
GAZİ ÜNİVERSİTESİ
Gazi
1926

İZİN VE İZİN GRUPLARINA DAYALI ANDROID KÖTÜCÜL YAZILIM TESPİT SİSTEMİ

Murat ÖNDER

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TEMMUZ 2019

Murat ÖNDER tarafından hazırlanan "İZİN VE İZİN GRUPLARINA DAYALI ANDROID KÖTÜCÜL YAZILIM TESPİT SİSTEMİ" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilgisayar Mühendisliği Ana Bilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Doç. Dr. İbrahim Alper DOĞRU

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

Başkan: Doç. Dr. Nursal ARICI

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……...

Üye: Dr. Öğr. Üyesi Abdullah Talha KABAKUŞ

Bilgisayar Mühendisliği Ana Bilim Dalı, Düzce Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……...

Tez Savunma Tarihi: 19/07/2019

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……..……

Prof. Dr. Sena YAŞYERLİ

Fen Bilimleri Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Murat ÖNDER
19/07/2019

İZİN VE İZİN GRUPLARINA DAYALI ANDROID KÖTÜCÜL YAZILIM TESPİT SİSTEMİ

(Yüksek Lisans Tezi)

Murat ÖNDER

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Temmuz 2019

ÖZET

Mobil cihazların en iyi performansla çalışması için optimize edilmiş olan Android işletim sisteminde çeşitli görevlerin yerine getirilebilmesi için cihaz üzerine çeşitli uygulamalar kurulmaktadır. Ancak mobil cihazın etkinliğini artırma amaçlı uygulamaların yanı sıra çok sayıda kötücül uygulama da geliştirilmekte ve uygulama mağazalarına yüklenmektedir. Çeşitli güvenlik şirketlerinin yayınladığı raporlarda Android işletim sistemine yönelik kötücül yazılım miktarı milyonlarla ifade edilmektedir. Kullanıcıların uygulamaların getirebileceği riskler ve korunma yöntemleri konusunda yeterli bilgiye ve bilince sahip olmamaları ise kullandıkları mobil cihazlara kötücül yazılım bulaşma riskini çok büyük oranda artırmaktadır. Bu nedenle kötücül yazılımların kullanıcılara ulaşmadan önce tespit edilip engellenmeleri, mobil cihazlara yüklenen kötücül yazılımların ise cihaza ve kullanıcıya zarar vermelerinin ve hassas verileri sızdırmalarının önlenmesi büyük önem taşımaktadır. Kötücül yazılımların mobil cihazlara yüklenmeden önce incelenerek tespit edilmesi amacı doğrultusunda izin ve izin gruplarına dayalı web tabanlı Android uygulama analiz sistemi olan ANUAS geliştirilmiştir. ANUAS, statik analiz yöntemlerini kullanarak incelenen uygulamaların manifest ve kod izinlerini çıkarmakta, bu izinlerden ikili ve üçlü izin grupları oluşturmaktadır. Ardından bu izinler ile izin gruplarının kötücül ve iyicil uygulamalarda kullanılma oranlarına göre risk puanlarını hesaplamakta ve tüm ayrı izin veya izin grubu risk puanlarını toplayarak incelenen uygulamanın toplam risk puanlarını bulmaktadır. ANUAS'ın etkinliğini değerlendirmek maksadıyla 3888 iyicil ve 3888 kötücül uygulama ile eğitilmiş, ardından 1666 iyicil ve 1666 kötücül uygulama ile sistem test edilmiştir. Yapılan testlerin sonucunda ANUAS en iyi doğruluk performansında kötücül uygulamaları %96,19 doğruluk, %95,50 hassasiyet ve %96,88 özgüllük oranıyla tespit etmiştir.

Bilim Kodu	:	92429
Anahtar Kelimeler	:	Android izinleri, İzin grupları, Android kötücül yazılımı, Kötücül yazılım tespiti, Statik analiz, Mobil güvenlik
Sayfa Adedi	:	119
Danışman	:	Doç. Dr. İbrahim Alper DOĞRU

ANDROID MALWARE DETECTION SYSTEM BASED ON PERMISSIONS AND PERMISSION GROUPS

(M. Sc. Thesis)

Murat ÖNDER

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

July 2019

ABSTRACT

In Android operating system which is optimized for the best performance of mobile devices, various applications are installed on the device to perform various tasks. However, in addition to applications aimed at increasing the efficiency of the mobile devices, a number of malicious applications are also being developed and installed in application stores. The reports issued by various security companies indicate the amount of malicious software for the Android operating system in millions. The fact that the users do not have sufficient knowledge and awareness about the risks and protection methods that applications may bring substantially increases the risk of infection of malicious software to the mobile devices they use. Therefore, it is of utmost importance that malicious software is detected and blocked before it reaches users and that malicious software installed on mobile devices is prevented from damaging the device as well as the user and leaking sensitive data. In accordance with the purpose of analyzing and detecting malicious applications before they are installed on mobile devices, ANUAS, a web-based Android application analysis system based on permission and permission groups, was developed. ANUAS extracts the manifest and code permissions of the examined applications using static analysis methods and creates binary and triple permission groups from these permissions. Then it calculates the risk scores according to the usage rates of these permissions and permission groups in malicious and benign applications and finds the total risk scores of the examined application by adding all of the separate permission or permission group risk scores. In order to evaluate the effectiveness of the ANUAS, it was trained with 3888 benign and 3888 malicious applications and subsequently it was tested with 1666 benign and 1666 malicious applications. In the tests, ANUAS detected malicious applications with 96,19% accuracy, 95,50% sensitivity and 96,88% specificity rate at its maximum accuracy performance.

Science Code	:	92429
Key Words	:	Android permissions, Permission groups, Android malware, Malware detection, Static analysis, Mobile security
Page Number	:	119
Supervisor	:	Assoc. Prof. Dr. İbrahim Alper DOĞRU

TEŞEKKÜR

Yüksek lisans sürecinde beni çalışmaktan çok büyük zevk aldığım bir alana yönlendiren, tüm süreç boyunca katkıları ve eleştirileriyle bana her zaman yol gösteren ve destek veren, en zorlandığım anlarda beni tekrar tekrar motive ederek ilerlememi sağlayan çok değerli hocam ve danışmanım Doç. Dr. İbrahim Alper DOĞRU'ya çok teşekkür ederim.

Yüksek lisans eğitimim boyunca bana karşı sevgisini, sabrını ve desteğini sürekli hissettiğim biricik eşim Bahar ÖNDER ile tüm hayatım ve eğitim sürecim boyunca maddi manevi her türlü desteklerini gördüğüm babam Hasan ÖNDER, annem Hasibe ÖNDER ve kardeşim Saliha ÖNDER'e de tüm kalbimle sonsuz teşekkür ederim.

Ayrıca bünyesinde görev yapmaktan çok büyük mutluluk duyduğum Sahil Güvenlik Komutanlığı ailesine ve özellikle yüksek lisans eğitimine başlamamda ve tamamlamamda büyük desteklerini gördüğüm amirlerim Mustafa YILMAZ Albay ile Mehmet Reşit ORMANOĞLU Albaya, değerli dostum Arda GÜLERER'e ve tüm çalışma arkadaşlarıma en içten dileklerimle teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
ms	Milisaniye

Kısaltmalar	**Açıklamalar**
API	Application Programming Interface (Uygulama Programlama Arayüzü)
APK	Android Package (Android Paketi)
ART	Android Runtime (Android Çalıştırma Ortamı)
CEO	Chief Executive Officer (İcra Kurulu Başkanı)
DEX	Dalvik Executable (Dalvik Çalıştırılabilir Dosya)
ICC	Inter-Component Communication (Bileşenler Arası İletişim)
IPC	Interprocess Communication (İşlemler Arası İletişim)
LKM	Loadable Kernel Module (Yüklenebilir Kernel Modülü)
OBB	Opaque Binary Blob (Opak İkili Büyük Nesne)
SDK	Software Development Kit (Yazılım Geliştirme Kiti)
SHA	Secure Hash Algorithm (Güvenli Özetleme Algoritması)
SMS	Short Message Service (Kısa Mesaj Hizmeti)
URL	Uniform Resource Locator (Tekdüzen Kaynak Konum Belirleyicisi)
XMPP	Extensible Messaging and Presence Protocol (Genişletilebilir Mesajlaşma ve Varlık Protokolü)

1. GİRİŞ

Mobil cihazların görevlerini yerine getirebilmesini ve son kullanıcılarla etkileşime geçebilmesini sağlamak amacıyla işletim sisteminin üzerine çeşitli uygulamalar kurulmaktadır. Ancak yapılan araştırmalara göre uygulama mağazalarında sunulan yüksek miktardaki uygulama arasında çok fazla kötücül yazılım da bulunmakta ve kötücül yazılım miktarı hızla artmaktadır [1, 2]. Kötücül yazılımlar, kullanıcıların sistem üzerindeki hassas bilgilerini ele geçirmek, bunları yetkisiz kişilere göndermek, sisteme ya da sistem üzerindeki dosyalara zarar vermek, sistem kaynaklarını sömürerek belirli işlemler yapmak gibi kullanıcılar tarafından istenmeyen çeşitli faaliyetler gerçekleştirerek kullanıcılara doğrudan veya dolaylı olarak zarar veren yazılımlardır. Genellikle kullanıcıların bilgisi ve onayı olmadan ya da kullanıcıları yanıltarak kullandıkları sisteme bulaşırlar ve eğer sistem bir ağa bağlıysa ağ üzerinden erişebildikleri diğer sistemlere yayılmaya çalışırlar [3]. G DATA firması tarafından 2018 yılı üçüncü çeyrek sonu itibariyle 3,2 milyon yeni mobil kötücül uygulama örneği tespit edildiği açıklanmıştır [4]. McAfee Labs firması tarafından yayınlanan 2019 yılı birinci çeyrek mobil tehdit raporunda ise 2018 yılı sonu itibariyle toplamda 30 milyondan fazla mobil kötücül yazılım olduğu açıklanmıştır [5].

Pulse Secure firması tarafından yapılan araştırmaya göre geliştirilen mobil kötücül yazılımların %97'si Android işletim sistemini hedef almaktadır. Firma tarafından bunun nedeni Android'in mevcut mobil cihaz platformları arasında en düşük cihaza giriş güvenliğini sağlaması olarak açıklanmıştır. Apple iOS işletim sisteminin detaylı bir ön inceleme süreci ve sıkı hizmet şartları geliştirilen kötücül yazılımların Apple iOS uygulama mağazasına girişini oldukça zorlaştırmaktadır. Buna karşın Android açık bir ekosistemde geliştirilmekte, resmi uygulama mağazası dışında üçüncü taraf uygulama mağazalarından da uygulama yüklenmesine izin vermekte ve mobil cihaz kullanıcılarının çok büyük çoğunluğu tarafından kullanılmaktadır. Bu durum kötücül yazılım geliştirenleri büyük oranda Android işletim sistemine yönlendirmektedir [6]. F-Secure firması tarafından yayınlanan raporda da 2016 yılı sonu itibariyle Android işletim sistemine yönelik olarak geliştirilen mobil kötücül yazılım sayısının 19 milyonu geçtiği ve geliştirilen tüm mobil kötücül yazılımların %99'unun Android işletim sistemini kullanan cihazları hedef aldığı açıklanmıştır. Firma bunun nedeni olarak çok geniş bir yelpazede Android cihazı bulunmasını ve mobil

uygulamaların dağıtımı için diğer mobil cihaz platformlarına göre daha açık bir sistem sunmasını göstermektedir [7].

Android işletim sisteminde uygulamaların çalışma sırasında hangi kaynaklara erişebileceklerini ve hangi işlemleri yapabileceklerini belirlemek amacıyla izin tabanlı güvenlik politikası kullanılmaktadır. Uygulamanın çalışması için gereken izinler uygulama geliştiricileri tarafından AndroidManifest.xml dosyası içinde belirtilmelidir. Android işletim sisteminde çok sayıda izin olmasına rağmen izin tabanlı güvenlik politikasının daha çok insan kaynaklı nedenlerle kötücül yazılımlara karşı güvenliği tam olarak sağlayamadığı çeşitli araştırmacılar tarafından yapılan çalışmalarda ortaya konmuştur. Örnek olarak Fang ve diğerleri yaptıkları çalışmada izin politikası ile ilgili olarak izinlerin kabaca bölümlenmesi, yeteneksiz izin yöneticileri, yetersiz izin dokümantasyonu, aşırı izin talebi, izin yükseltme saldırısı ve kontrol zamanı kullanım zamanı saldırısı alanlarında sorunlar bulunduğunu belirlemişler ve bu sorunlu alanlarla ilgili olarak literatürde yapılan çalışmaları özetleyerek çeşitli istatistikler sunmuşlardır [8].

Bu tez çalışmasının amacı Android işletim sisteminde bulunan izin tabanlı güvenlik politikasını kullanarak kötücül yazılımları mobil cihaza yüklenmeden önce inceleyip tespit eden bir sistem geliştirmektir. Bu amaç doğrultusunda geliştirilen mobil uygulamaların talep ettikleri ve kod içerisinde kullandıkları izinleri ortaya çıkaran ve bunları statik analiz yöntemleriyle inceleyip değerlendiren yeni bir yaklaşım ortaya konmuş ve bu yaklaşıma dayanan web tabanlı Android Uygulama Analiz Sistemi (ANUAS) geliştirilmiştir. ANUAS, mobil uygulama dosyalarının sisteme yüklenmesini ya da sistem tarafından internetten indirilmesini sağlayan bir web arayüzü, inceleme faaliyetini yürüten bir inceleme yazılımı ve sonuçları tutan bir veri tabanından oluşmaktadır.

Tez çalışmasının 2. bölümünde Android işletim sistemine ve Android'e yönelik olarak geliştirilen kötücül yazılımların özelliklerine ilişkin genel bilgiler verilmiştir. 3. bölümde Android işletim sistemine yönelik kötücül yazılımlara karşı mücadele kapsamında kötücül yazılımları tespit etme ya da kötücül yazılımlardan korunma amacıyla yapılan çalışmalar incelenmiştir. 4. bölümde ANUAS'ın çalışma mantığı, web arayüzü ve veri tabanı ayrıntılı bir şekilde anlatılmıştır. 5. bölümde yapılan testler ve elde edilen bulgular açıklanmış ve 6. bölümde sonuç ve önerilere yer verilmiştir.

2. ANDROID İŞLETİM SİSTEMİ

Bu bölümde Android işletim sisteminin tarihsel gelişimi ve mimarisi, Android uygulamalarının yapısı ve geliştirilme süreci, Android işletim sisteminin izin tabanlı güvenlik politikası ile kullanılan güvenlik sistemlerine ilişkin genel bilgiler sunulacaktır.

2.1. Android İşletim Sisteminin Tarihsel Gelişimi

Hâlihazırda Apple iOS ile birlikte en yaygın kullanılan iki mobil cihaz işletim sisteminden biri olan Android işletim sistemi ilk olarak 2003 yılında kurulan Android Inc. tarafından geliştirilmeye başlanmış, 2005 yılında Android Inc. şirketi Google şirketi tarafından satın alınmıştır [9]. 5 Nisan 2007 tarihinde mobil cihazlar için açık ve kapsamlı bir platform geliştirmek üzere Google'ın öncülüğünde aralarında HTC, LG, Motorola, Samsung ve Sony gibi cihaz üreticileri, Sprint Nextel, T-Mobile, China Mobile ve Telecom Italia gibi ağ işletmecileri, Intel, Qualcomm, NVIDIA Corporation ve Texas Instruments gibi donanım üreticilerinin bulunduğu 34 firma tarafından Open Handset Alliance konsorsiyumu kurulmuştur [10, 11]. Linux tabanlı açık kaynak kodlu bir işletim sistemi olan Android hâlihazırda 84 teknoloji ve mobil cihaz şirketinin üye olduğu bu konsorsiyum tarafından geliştirilmektedir [12].

Android işletim sisteminin ilk ticari sürümü olan 1.0 sürümü T-Mobile G1 olarak da bilinen HTC Dream telefon üzerinde 23 Eylül 2008 yılında piyasaya sürülmüştür. Android, mobil cihaz piyasasındaki pazar payını hızla artırarak 2010 yılından itibaren hem Amerika'da hem de tüm dünyada liderliği ele geçirmeye başlamıştır [9, 13, 14]. IDC firmasına göre 2018 yılında %85,1'lik pazar payı ile en yaygın kullanılan mobil cihaz işletim sistemi olmayı sürdüren Android'in gelecek yıllarda pazar payını daha da artırması beklenmektedir [15]. Android kıdemli yöneticisi Stephanie Cuthbertson 07 Mayıs 2019 tarihinde Google I/O 2019 yıllık geliştirici konferansında Android işletim sisteminin toplamda 2,5 milyardan fazla aktif cihazda kullanılmakta olduğunu duyurmuştur [16, 17]. Daha öncesinde Google I/O 2017 yıllık geliştirici konferansında ise Google CEO'su Sundar Pichai tarafından Android işletim sisteminin toplamda 2 milyardan fazla aktif cihazda kullanılmakta olduğu açıklanmıştır [18]. Bu açıklamalar iki yılda Android işletim sisteminin yarım milyardan fazla aktif cihaza daha kurulduğunu ve hızla daha fazla cihaza yayıldığını göstermektedir.

Android işletim sisteminin hâlihazırdaki son ana sürümü 6 Ağustos 2018'de piyasaya çıkartılan Android 9 (Pie) sürümüdür. 2019 yılı Mart ayında Android 10 (Q) Beta 1 sürümü test kullanıcılarına sunulmaya başlanmış olup nihai sürümün Ağustos ayı içerisinde piyasaya çıkartılması beklenmektedir [19]. Android sürümlerinin (her farklı API'deki ilk Android sürümü) çıkış tarihleri Çizelge 2.1'de sunulmuştur.

Çizelge 2.1. Android sürümlerinin çıkış tarihleri [20]

Android Sürümü	İsim	API	Çıkış Tarihi
1.0	-	1	23 Eylül 2008
1.1	Petit Four	2	9 Şubat 2009
1.5	Cupcake	3	27 Nisan 2009
1.6	Donut	4	15 Eylül 2009
2.0	Eclair	5	26 Ekim 2009
2.0.1		6	3 Aralık 2009
2.1		7	12 Ocak 2010
2.2	Froyo	8	20 Mayıs 2010
2.3	Gingerbread	9	6 Aralık 2010
2.3.3		10	9 Şubat 2011
3.0	Honeycomb	11	22 Şubat 2011
3.1		12	10 Mayıs 2011
3.2		13	15 Temmuz 2011
4.0	Ice Cream Sandwich	14	18 Ekim 2011
4.0.3		15	16 Aralık 2011
4.1	Jelly Bean	16	9 Temmuz 2012
4.2		17	13 Kasım 2012
4.3		18	24 Temmuz 2013
4.4	KitKat	19	31 Ekim 2013
4.4W [1]		20	25 Haziran 2014
5.0	Lollipop	21	12 Kasım 2014
5.1		22	9 Mart 2015
6.0	Marshmallow	23	5 Ekim 2015
7.0	Nougat	24	22 Ağustos 2016
7.1		25	4 Ekim 2016
8.0	Oreo	26	21 Ağustos 2017
8.1		27	5 Aralık 2017
9	Pie	28	6 Ağustos 2018
10	Q	29	Ağustos 2019'da çıkartılması beklenmektedir.

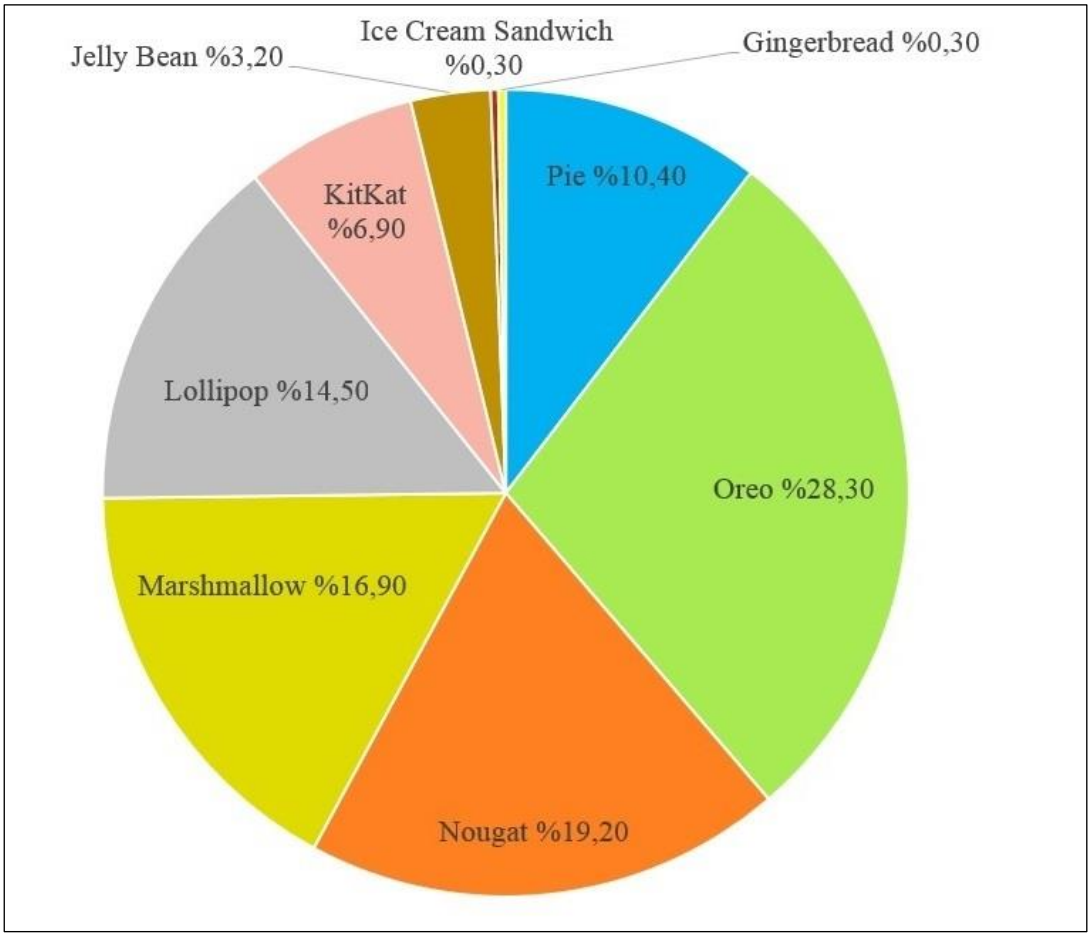
2019 yılı Mayıs ayı itibariyle Android sürümlerinin sürüm numaralarına göre kullanım oranları Çizelge 2.2'de sunulmuştur. Kullanım oranı %0,1'in altında kalan sürümler bu çizelgeye dâhil edilmemiştir.

[1] Android 4.4W (API 20) sürümü akıllı saatler gibi Android Wear cihazlarına (giyilebilir cihazlar) özel olarak çıkartılmış bir sürümdür. Android 4.4 (API 19) sürümünün üzerine giyilebilir cihaz eklentileri ilave edilmiştir.

Çizelge 2.2. Android sürümlerinin sürüm numaralarına göre kullanım oranları [21]

Android Sürümü	İsim	API	Kullanım Oranı
2.3.3 – 2.3.7	Gingerbread	10	%0.3
4.0.3 – 4.0.4	Ice Cream Sandwich	15	%0.3
4.1.x	Jelly Bean	16	%1.2
4.2.x		17	%1.5
4.3		18	%0.5
4.4	KitKat	19	%6.9
5.0	Lollipop	21	%3.0
5.1		22	%11.5
6.0	Marshmallow	23	%16.9
7.0	Nougat	24	%11.4
7.1		25	%7.8
8.0	Oreo	26	%12.9
8.1		27	%15.4
9	Pie	28	%10.4

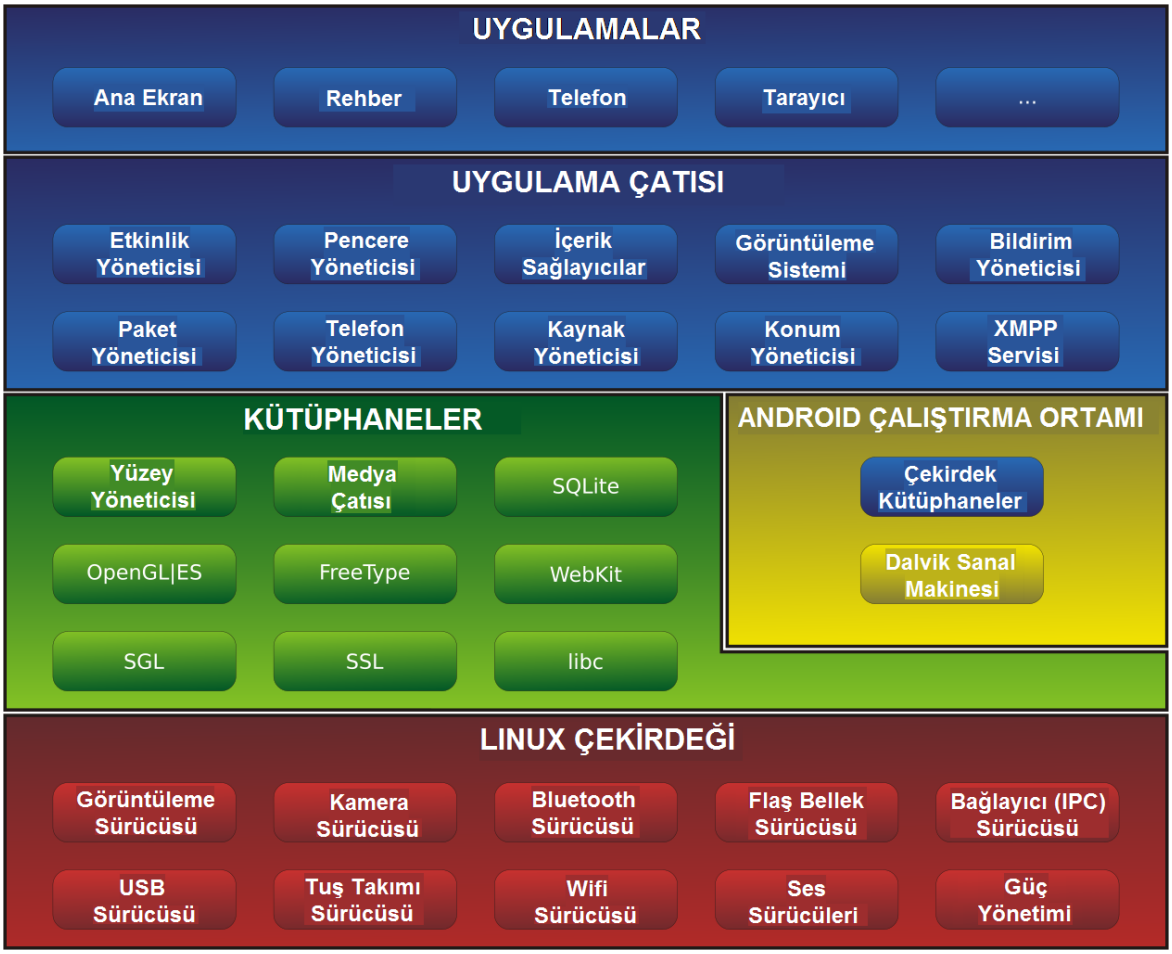
Android sürümlerinin sürüm adlarına göre kullanım oranları Şekil 2.1'de gösterilmiştir.



Şekil 2.1. Android sürümlerinin sürüm adlarına göre kullanım oranları [21]

2.2. Android İşletim Sistemi Mimarisi

Android işletim sistemi Linux çekirdeği (kernel) üzerine inşa edilmiş ve temel olarak akıllı telefonlar, tabletler vb. dokunmatik cihazlar için geliştirilmiş çok katmanlı bir işletim sistemidir. Şekil 2.2'de Android işletim sisteminin mimarisi sunulmuştur.



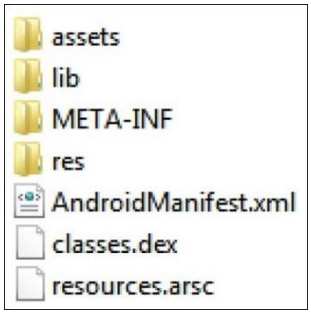
Şekil 2.2. Android işletim sistemi mimarisi [22]

Android işletim sisteminin en alt katmanında mobil cihazın ekran, kamera, pil, wifi, bluetooth, USB, ses vb. donanım bileşenleri ile iletişimi sağlayan özelleştirilmiş bir Linux çekirdeği bulunmaktadır. Bir üst katmanda çeşitli kütüphaneler ile Android Çalıştırma Ortamı ve üstünde uygulama çatısı katmanı bulunmaktadır. En üst katmanda ise kullanıcı ile etkileşimin gerçekleştirildiği ve akıllı cihazın görevlerini yapmasını sağlayan çağrı uygulaması, mesajlaşma uygulaması, internet tarayıcı, kamera uygulaması gibi çeşitli uygulamalar bulunmaktadır. Android işletim sisteminde cihaz üreticileri ya da bağımsız geliştiriciler tarafından geliştirilen her uygulama, ayrı bir çalıştırma ortamı örneğinde çalışır,

böylece uygulamalar birbirinden izole edilmiş olur. Android işletim sisteminin 4.4 KitKat sürümüne kadar çalıştırma ortamı olarak sadece Java sanal makinesinin optimize edilmiş bir hali olan Dalvik sanal makinesi kullanılmıştır. Android 4.4 KitKat sürümüne Dalvik sanal makinesinin yanı sıra teknoloji ön izlemesi maksadıyla ART çalıştırma ortamı da eklenmiş, 5.0 Lollipop sürümünden itibaren Dalvik sanal makinesinin yerini ART çalıştırma ortamı almıştır [23, 24]. Uygulamaların bayt kodunun sık olarak kullanılan parçalarını dinamik olarak makine koduna derleyen Dalvik'in aksine ART uygulamaların tüm kodunu kurulum sırasında makine koduna derlemektedir. Bu işlem çalıştırma etkinliğini artırmakta ve güç tüketimini azaltmaktadır. ART aynı zamanda uygulamaların daha hızlı çalışmasını sağlamakta, daha iyi bellek ayırma ve çöp toplama performansı sergilemekte ve yeni uygulama hata ayıklama özellikleri sunmaktadır. Geçmişle uyumluluğu sağlamak için ART, Dalvik ile aynı şekilde APK dosyaları içerisinde dex formatında sunulan Dalvik bayt kodunu kullanmaktadır [25].

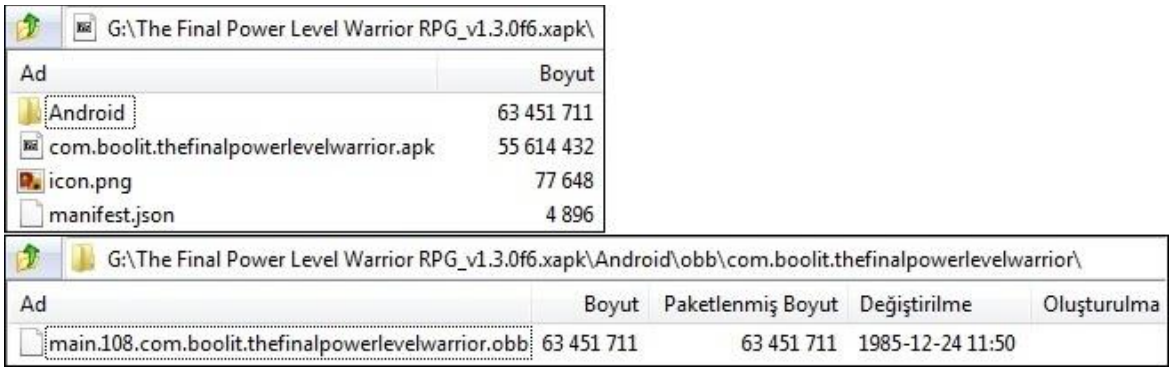
2.3. Android Uygulamaları

Android işletim sisteminde çalışacak uygulamalar önce Android Application Programming Interface (API) kullanılarak Java dilinde yazılır, ardından yazılan kod Java bayt kod olarak derlenir ve .class uzantılı dosyalar oluşur. Daha sonra dx derleyicisi ile bütün Java .class dosyaları Dalvik bayt kod olarak derlenir, .dex formatına dönüştürülür ve tek bir classes.dex dosyasında ya da classes2.dex, classes3.dex vb. isimlendirmeyle birden fazla dex dosyasında toplanır. Uygulama kodunun yanı sıra uygulamanın kullandığı kaynaklar ile uygulamanın çalışması için gereken izinlerin tanımlandığı AndroidManifest.xml dosyası tek bir APK dosyası içine toplanır. Son olarak oluşturulan APK dosyası geliştirici tarafından imzalanarak kullanıma sunulur [26]. Şekil 2.3'te örnek bir APK dosyası içeriği gösterilmektedir.



Şekil 2.3. Örnek bir APK dosyası içeriği

Android uygulama dosyaları APK dışında XAPK formatında da olabilir. XAPK dosyaları standart bir APK dosyasının dışında uygulamanın kullandığı diğer grafik, medya dosyaları ile verileri bulunduran bir OBB dosyasını, uygulama ikonunu ve manifest.json isimli bilgi dosyasını içerirler. XAPK dosyaları Android işletim sisteminin resmi uygulama mağazası olan Google Play'de desteklenmemekte olup sadece üçüncü taraf uygulama mağazalarından indirilebilmektedir. Ayrıca XAPK formatı Google tarafından desteklenmediği için cihaz üzerine doğrudan Android tarafından kurulamamaktadır. XAPK formatındaki uygulamaları cihaza kurabilmek için cihaz üzerine öncelikle XAPK yükleyici bir uygulama yüklenmesi gereklidir [27, 28]. Şekil 2.4'te örnek bir XAPK dosyası içeriği gösterilmektedir.

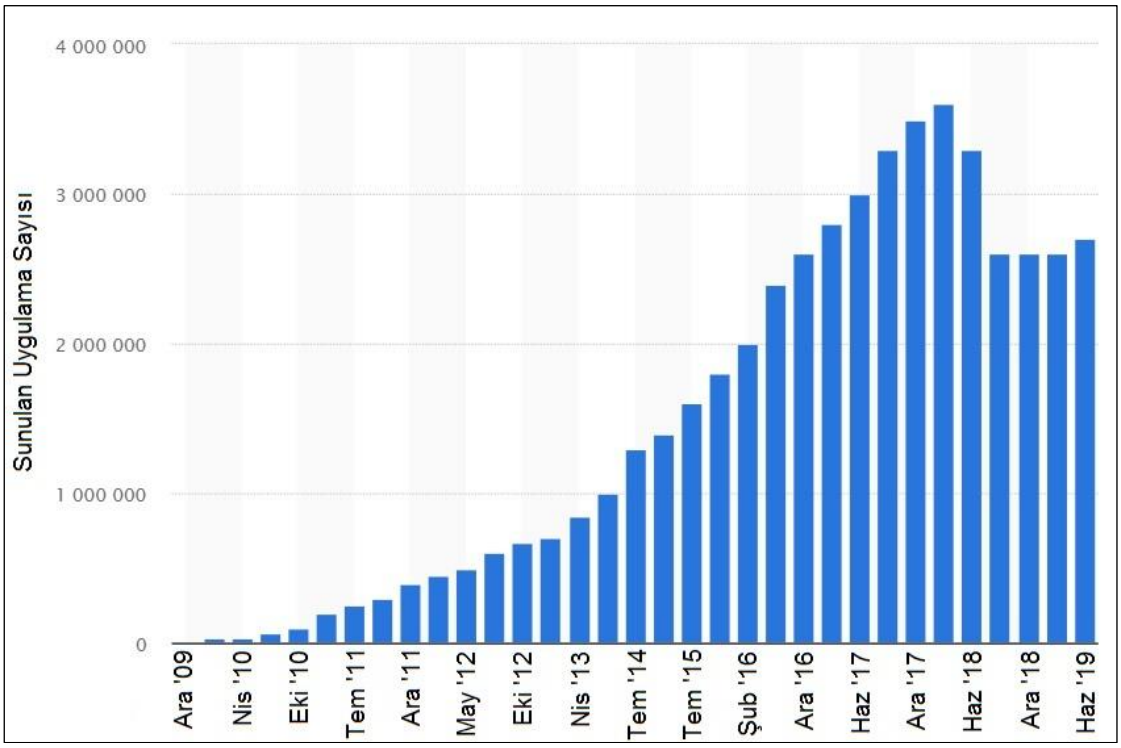


Şekil 2.4. Örnek bir XAPK dosyası içeriği

XAPK formatının ortaya çıkış sebebi Google Play'in 2015 yılına kadar APK boyutu olarak en fazla 50 MB sınırı koymasıdır. Bu sınır 2015 yılında 100 MB'a yükseltilmiştir [29]. Ayrıca mağazaya yüklenen APK dosyaları için her biri en fazla 2GB olabilen iki taneye kadar APK genişleme paketi desteklenmektedir. Bu paketler yükleyen kişinin istediği herhangi bir formatta (zip, pdf, mp4 vb.) olabilir ancak yüklendikten sonra Google Play tarafından .obb uzantısı ile bitecek şekilde yeniden isimlendirilirler. Bir uygulama cihaza kurulurken Google Play genişleme paketlerini indirebilirse APK dosyası ile birlikte indirir. Eğer indiremezse sadece APK paketini indirir ve uygulamayı kurar. Cihaza kurulan uygulama çalışmaya başladıktan sonra genişleme paketleri uygulama tarafından indirilmeye başlanır. Bu nedenle genişleme paketi kullanarak uygulama geliştirenlerin uygulama içerisine genişleme paketini kontrol etme ve gerekiyorsa indirme kodunu eklemeleri de gereklidir [30]. Bunun dışında Google I/O 2018 konferansında Android uygulamalarının dağıtımı için Android App Bundle isimli yeni bir format tanıtılmıştır. Bu format ile 150

MB'a kadar APK uygulamalarının tek bir paket halinde dağıtımı mümkün hale gelmiştir [31].

Android işletim sistemini kullanan cihazlara yüklenecek uygulamaların kullanıcılara ulaştırılmasını kolaylaştırmak amacıyla pek çok uygulama mağazası oluşturulmuştur. Android işletim sistemi için bu amaç doğrultusunda 2008 yılında Android Market uygulama mağazasını hizmete sunmuş, 2012 yılında Android Market'in yanı sıra Google Music ve Google eBookstore mağazaları Google Play ismi altında birleştirilmiştir [32]. 2019 yılı Temmuz ayı itibariyle Google Play uygulama mağazasında 2,7 milyondan fazla uygulama bulunmaktadır [33, 34]. Şekil 2.5'te Google Play'deki uygulamaların sayısının 2009-2019 yılları arasındaki değişimi gösterilmektedir.



Şekil 2.5. Google Play'deki uygulama sayısının değişimi [34]

Google Play haricinde uygulamalar ayrıca Samsung Galaxy Store gibi cihaz üreticilerinin resmi uygulama mağazalarından da indirilebilir. Bunların dışında Amazon Appstore, APKMirror, APKPure, Aptoide gibi pek çok üçüncü taraf uygulama mağazası da bulunmaktadır [35]. Uygulamalar bu mağazalardan da APK ya da XAPK formatlarında indirilebilir. Ayrıca cihaza yüklenecek uygulama APK ya da XAPK dosyası olarak USB, Bluetooth vb. yöntemlerle de cihaza taşınabilir ve yüklenebilir.

2.4. Android İşletim Sisteminde Güvenlik

Bu bölümde öncelikle Android işletim sisteminde kullanılmakta olan izin tabanlı güvenlik politikası, Android sürümlerine göre bu politikanın değişimi ve izin türleri, ardından Google tarafından Google Play uygulama mağazasında ve Android cihazlar üzerinde kullanılan güvenlik sistemleri ve bu sistemlere ilişkin problemler açıklanacaktır.

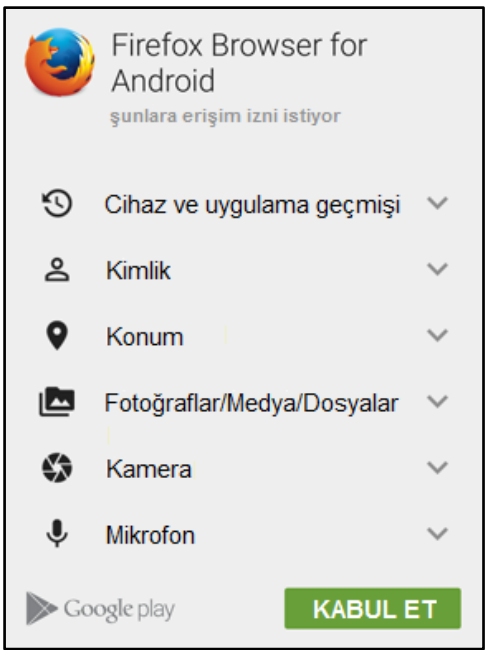
2.4.1. İzin tabanlı güvenlik politikası

Uygulamaların çalışma sırasında hangi kaynaklara erişebileceklerini ve hangi işlemleri yapabileceklerini belirlemek amacıyla Android işletim sisteminde izin tabanlı güvenlik politikası kullanılmaktadır. Bu politikaya göre herhangi bir uygulamanın çalışması için gereken izinler uygulama geliştiricileri tarafından AndroidManifest.xml dosyası içinde <uses-permission> ya da <uses-feature> etiketleriyle belirtilmelidir. AndroidManifest.xml dosyası içerisine talep edilen erişim izinlerinin yanı sıra uygulamanın paket adı, en düşük SDK sürümü, en yüksek SDK sürümü ve hedeflenen SDK sürümü, kullandığı yazılımsal ya da donanımsal kaynaklar gibi uygulamanın çalışmasına ilişkin pek çok bilgi eklenebilir. Uygulamalar tarafından varsayılan Android izinlerinin dışında yeni izinler de bu dosya içerisinde tanımlanabilir [36]. AndroidManifest.xml dosyası içerisinde talep edilen izinler bu tez çalışmasında manifest izinleri olarak adlandırılacaktır. Şekil 2.6'da örnek bir AndroidManifest.xml dosyasının izin talebi bölümü gösterilmektedir.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?><manifest xmlns:android=
"http://schemas.android.com/apk/res/android" android:compileSdkVersion="28" android:compileSdkVersionCodename="9"
android:installLocation="auto" package="org.videolan.vlc" platformBuildVersionCode="28" platformBuildVersionName="9">
    <permission android:name="org.videolan.vlc.permission.READ_EXTENSION_DATA" android:protectionLevel="normal"/>
    <permission android:name="org.videolan.vlc.permission.BIND_DATA_CONSUMER" android:protectionLevel="normal"/>
    <uses-permission android:name="org.videolan.vlc.permission.READ_EXTENSION_DATA"/>
    <uses-permission android:name="org.videolan.vlc.permission.BIND_DATA_CONSUMER"/>
    <uses-permission android:name="android.permission.VIBRATE"/>
    <uses-permission android:name="android.permission.WRITE_SETTINGS"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
    <uses-permission android:name="android.permission.RECORD_AUDIO"/>
    <uses-permission android:name="android.permission.FOREGROUND_SERVICE"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.WAKE_LOCK"/>
    <uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
    <uses-permission android:name="android.permission.BLUETOOTH"/>
    <uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"/>
    <uses-permission android:name="com.android.providers.tv.permission.READ_EPG_DATA"/>
    <uses-permission android:name="com.android.providers.tv.permission.WRITE_EPG_DATA"/>
```

Şekil 2.6. Örnek bir AndroidManifest.xml dosyasının izin talebi bölümü

Android 6.0 Marshmallow sürümünden önceki sürümlerde herhangi bir uygulama kurulmadan önce talep edilen tehlikeli izinler gruplar halinde kullanıcıya gösterilmekte ve kullanıcının talep edilen izin gruplarına onay vermesi beklenmektedir. Android "hepsi ya da hiçbiri" şeklinde bir izin onay politikası uygulamaktadır. Kullanıcı izinlerin ya hepsini onaylar, böylece uygulama cihaza kurulur ya da kullanıcı izin talebini reddeder, uygulamanın kurulumu sonlandırılır, uygulama cihaza kurulmaz. Kullanıcı kurulum sırasında izinleri tek tek seçememektedir, ya hepsine birden izin vermeli ya da hepsini birden reddetmelidir. Ayrıca uygulama kurulduktan sonra verilen izinleri iptal etme imkânı da bulunmamaktadır. [37]. Şekil 2.7'de örnek bir kurulum sırasında izin talebi gösterilmektedir.



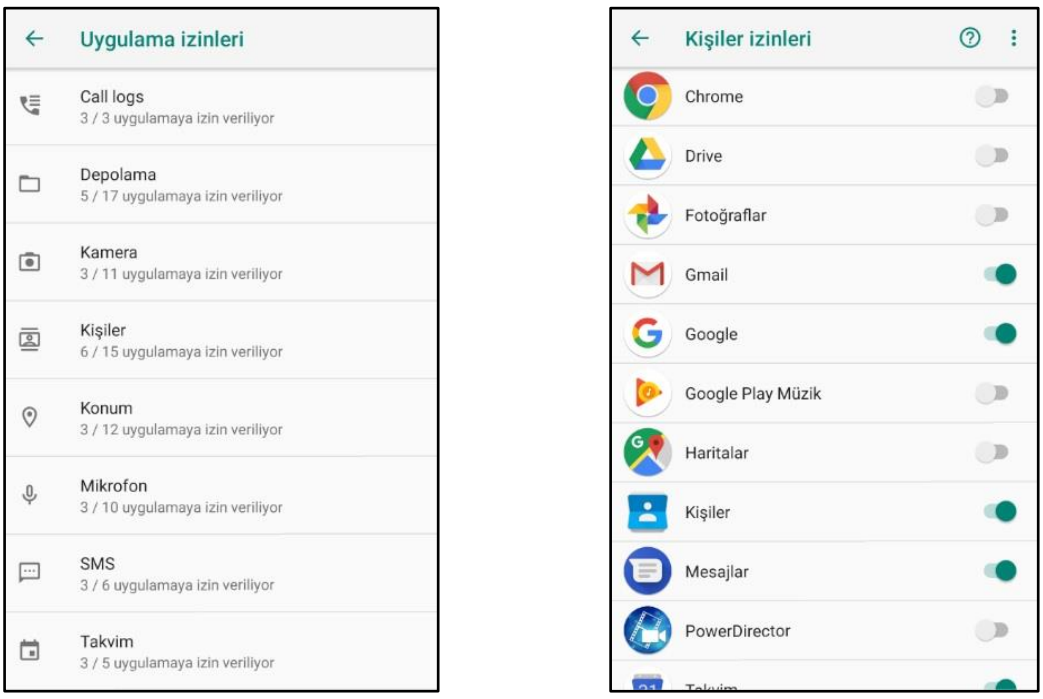
Şekil 2.7. Örnek bir kurulum sırasında izin talebi

Android işletim sisteminin izin tabanlı güvenlik politikası Android 6.0 Marshmallow sürümüyle birlikte büyük bir değişikliğe uğramıştır. Bu sürümle birlikte kurulum öncesinde kullanıcılardan herhangi bir izin talep edilmeden uygulamalar kurulmaya ve uygulamanın çalışması sırasında belirli işlemler için kullanıcılardan izin talep edilmeye başlanmıştır [38]. Uygulama cihaz üzerindeki hassas bir kaynağa erişmek istediğinde Şekil 2.8a'da gösterildiği gibi kullanıcıdan erişim izni istenmektedir. Kullanıcı bu pencerede erişim izni verebilir ya da reddedebilir. Reddetmesi durumunda uygulama tekrar aynı kaynağa erişmek istediğinde Şekil 2.8b'de gösterildiği gibi "Tekrar sorma" seçeneğinin de bulunduğu bir pencerede kullanıcıdan tekrar erişim izni istenmektedir.

a b

Şekil 2.8. a) İlk izin talebi b) Sonraki izin talepleri

Android 6.0 Marshmallow sürümüyle birlikte kullanıcılardan uygulamaların çalışması sırasında erişim izni talep edilmesinin yanı sıra uygulamalara gruplar halinde ayrı ayrı izin verebilme ya da verilmiş olan izinleri gruplar halinde ayrı ayrı iptal edebilme imkânı da kullanıcılara sunulmuştur [38]. Şekil 2.9a'da Android işletim sisteminin Uygulama izinleri bölümü, Şekil 2.9b'de örnek olarak Kişiler kaynağının izinleri gösterilmektedir. Uygulama izinlerinde hangi izin grubu için kaç uygulamaya izin verildiği, herhangi bir izin grubuna girildiğinde ise bu izin grubu için hangi uygulamalara izin verildiği görülebilir ve farklı uygulamalar için izin durumu değiştirilebilir.



a b

Şekil 2.9. a) Uygulama izinleri b) Kişiler izinleri

Android işletim sisteminde varsayılan olarak tanımlı izinler her API ve her işletim sistemi sürümünde farklılık göstermektedir. Android geliştiricileri her API ve işletim sistemi güncellemesinde genellikle mevcut izinlerin üzerine pek çok yeni izin eklemektedir. Tez çalışması kapsamında Android Studio yazılımının bir bileşeni olan Android Emulator kullanılarak Android 2.3 (API 9) sürümünden Android 10 (API 29) sürümüne kadar olan tüm sürümlerin varsayılan izinleri ve izin sayıları çıkarılmıştır. Android Studio yazılımı Android işletim sisteminde çalışmak üzere uygulama geliştirenler tarafından kullanılması maksadıyla Android geliştiricileri tarafından piyasaya sürülen bir yazılım geliştirme ortamıdır [39]. Android Emulator ise Android için geliştirilen uygulamaların farklı Android ve API sürümleri ile çeşitli cihazlarda test edilebilmesi için geliştirilen bir simülasyon bileşenidir [40]. Tez süresi boyunca belirli aralıklarla farklı Android sürümlerinin x86 işletim sistemi imajlarıyla sanal cihazlar oluşturulmuş, bu cihazlar çalıştırılarak komut satırından "*adb shell pm list permissions -g*" komutu girilerek varsayılan olarak tanımlı izinler çıkarılmıştır. Android Emulator, 2019 yılı Temmuz ayı itibariyle API 10 ve API 15-29 arasındaki x86 imajlarını sunmakta olup diğer imajlar sistemden kaldırılmıştır. API 15 imajı ise sorunlu olup bu imajla kurulan sanal cihazlar çalıştırılamamaktadır. Bu nedenle 2019 yılı Temmuz ayında yapılan son izinler ve izin sayıları çıkarma çalışmasında sadece API 10 ile API 16-29 arasındaki sürümlerin izinleri güncellenebilmiştir. Güncelleme çalışmasında kullanılan Android Emulator sanal cihazları Şekil 2.10'da gösterilmektedir.

Tür	Ad	Play Store	Çözünürlük	API	Hedef	CPU/ABI	Boyut	Eylemler
	Nexus 5X API 10		1080 × 1920: 420dpi	10	Android 2.3.3 (Google APIs)	x86	2.7 GB	
	Nexus 5X API 16		1080 × 1920: 420dpi	16	Android 4.1 (Google APIs)	x86	2.7 GB	
	Nexus 5X API 17		1080 × 1920: 420dpi	17	Android 4.2 (Google APIs)	x86	2.7 GB	
	Nexus 5X API 18		1080 × 1920: 420dpi	18	Android 4.3 (Google APIs)	x86	2.7 GB	
	Nexus 5X API 19		1080 × 1920: 420dpi	19	Android 4.4 (Google APIs)	x86	2.9 GB	
	Nexus 5X API 21		1080 × 1920: 420dpi	21	Android 5.0 (Google APIs)	x86	3.1 GB	
	Nexus 5X API 22		1080 × 1920: 420dpi	22	Android 5.1 (Google APIs)	x86	3.3 GB	
	Nexus 5X API 23		1080 × 1920: 420dpi	23	Android 6.0 (Google APIs)	x86	2.7 GB	
	Nexus 5X API 24		1080 × 1920: 420dpi	24	Android 7.0 (Google APIs)	x86	1.3 GB	
	Nexus 5X API 25		1080 × 1920: 420dpi	25	Android 7.1.1 (Google APIs)	x86	1.2 GB	
	Nexus 5X API 26		1080 × 1920: 420dpi	26	Android 8.0 (Google APIs)	x86	1.3 GB	
	Nexus 5X API 27		1080 × 1920: 420dpi	27	Android 8.1 (Google APIs)	x86	1.2 GB	
	Nexus 5X API 28		1080 × 1920: 420dpi	28	Android 9.0 (Google APIs)	x86	1.2 GB	
	Nexus 5X API 29		1080 × 1920: 420dpi	29	Android 9.+ (Google APIs)	x86	838 MB	

Şekil 2.10. İzinleri belirlemede kullanılan Android Emulator sanal cihazları

Tez çalışması kapsamında bu yolla elde edilen Android 2.3 (API 9) ile Android 10 (API 29) arasındaki Android sürümlerinin izin sayıları Çizelge 2.3'te sunulmuştur. Akademik araştırmalar kapsamında talep edilmesi halinde tüm sürümlerin elde edilen tüm izinlerini ve her bir iznin farklı Android sürümlerindeki durumunu gösteren detaylı karşılaştırma dosyaları araştırmacılarla paylaşılabilir. Ayrıca Android geliştiricileri tarafından sunulan ve AndroiManifest.xml dosyası içerisinde talep edilebilecek varsayılan izinlerin bir bölümünü gösteren manifest izinleri sayfasından da izinlerin isimleri ve kullanım amaçlarına ilişkin önemli bilgiler elde edilebilir [41].

Çizelge 2.3. Android sürümlerinin izin sayıları

Android Sürümü	İsim	API	Çıkış Tarihi	İzin Sayısı
2.3	Gingerbread	9	6 Aralık 2010	151
2.3.7		10	21 Eylül 2011	252
3.0	Honeycomb	11	22 Şubat 2011	155
3.1		12	10 Mayıs 2011	155
3.2		13	15 Temmuz 2011	157
4.0	Ice Cream Sandwich	14	18 Ekim 2011	178
4.0.3		15	16 Aralık 2011	181
4.1.2	Jelly Bean	16	9 Ekim 2012	300
4.2.2		17	11 Şubat 2013	320
4.3.1		18	3 Ekim 2013	327
4.4.2	KitKat	19	9 Aralık 2013	383
5.0.2	Lollipop	21	19 Aralık 2014	416
5.1.1		22	21 Nisan 2015	439
6.0	Marshmallow	23	5 Ekim 2015	412
7.0	Nougat	24	22 Ağustos 2016	472
7.1.1		25	5 Aralık 2016	469
8.0	Oreo	26	21 Ağustos 2017	556
8.1		27	5 Aralık 2017	585
9	Pie	28	6 Ağustos 2018	643
10	Q	29	Ağustos 2019'da çıkartılması beklenmektedir.	734

Android işletim sisteminde talep edilebilecek izinler hâlihazırda Normal İzinler, Tehlikeli İzinler ve İmza İzinleri olarak üç ayrı koruma seviyesine ayrılmıştır. Bu koruma seviyeleri, izinlerin potansiyel risklerini ve herhangi bir uygulama bir izin talebinde bulunduğu zaman onay verilip verilmemesi konusunda Android işletim sisteminin izleyeceği prosedürü belirlemek amacıyla kullanılmaktadır [42].

Normal İzinler

Varsayılan koruma seviyesidir. Normal izinler diğer uygulamalara, sisteme ya da kullanıcıya zarar verme ihtimali düşük olan, izole edilmiş uygulama seviyesi özelliklere erişim için verilen izinlerdir. Bu gruptaki izinler uygulamanın kurulumu sırasında sistem tarafından otomatik olarak verilir, uygulamanın kurulumu ya da çalışması sırasında kullanıcıdan onay istenmez. Bu izinler Android işletim sistemi ayarlarında Uygulama İzinleri bölümünde görüntülenmez ve uygulama kurulduktan sonra iptal edilemez. Çizelge 2.4'te örnek olarak normal izinlerden bazıları gösterilmektedir.

Çizelge 2.4. Örnek normal izinler

İzin Adı
ACCESS_NETWORK_STATE
ACCESS_WIFI_STATE
BLUETOOTH
CHANGE_NETWORK_STATE
CHANGE_WIFI_STATE
INTERNET
NFC
RECEIVE_BOOT_COMPLETED
SET_ALARM
SET_WALLPAPER
VIBRATE

Tehlikeli İzinler

Tehlikeli izinler kullanıcıların kişisel bilgilerini içeren kaynaklara erişmek veya kullanıcı verilerini ya da diğer uygulamaların çalışmasını olumsuz etkileyebilecek eylemler gerçekleştirmek için gereken izinlerdir. Bu izinlerin kullanıcılara zarar verme potansiyeli olması nedeniyle bu izinler sistem tarafında otomatik olarak verilmez, kullanıcının izin talebini onaylaması beklenir. Kullanıcı talebi onaylayana kadar uygulama bu izne bağlı faaliyetini gerçekleştiremez.

Android işletim sisteminde tehlikeli izinler daha kolay yönetilebilmeleri maksadıyla çeşitli izin gruplarında toplanmıştır. Örnek olarak Android 9 (Pie) (API 28) sürümünde 11 grup

altında 36 adet tehlikeli izin bulunmaktadır. Çizelge 2.5'te Android 9 (Pie) (API 28) sürümünde bulunan tehlikeli izin grupları ve izinler gösterilmektedir.

Çizelge 2.5. Tehlikeli izin grupları ve izinler

İzin Grubu Adı	İzin Adı
ARABA BİLGİLERİ	CAR_FUEL
	CAR_MILEAGE
	CAR_VENDOR_EXTENSION
ARAMA KAYITLARI	PROCESS_OUTGOING_CALLS
	READ_CALL_LOG
	WRITE_CALL_LOG
DEPOLAMA	READ_EXTERNAL_STORAGE
	WRITE_EXTERNAL_STORAGE
KAMERA	CAMERA
KİŞİLER	GET_ACCOUNTS
	READ_CONTACTS
	WRITE_CONTACTS
KONUM	ACCESS_COARSE_LOCATION
	ACCESS_FINE_LOCATION
	CAR_SPEED
MİKROFON	RECORD_AUDIO
SMS	READ_CELL_BROADCASTS
	READ_SMS
	RECEIVE_MMS
	RECEIVE_SMS
	RECEIVE_WAP_PUSH
	SEND_SMS
TAKVİM	READ_CALENDAR
	WRITE_CALENDAR
TELEFON	ACCEPT_HANDOVER
	ACCESS_UCE_OPTIONS_SERVICE
	ACCESS_UCE_PRESENCE_SERVICE
	ANSWER_PHONE_CALLS
	CALL_PHONE
	READ_PHONE_NUMBERS
	READ_PHONE_STATE
	USE_SIP
	ADD_VOICEMAIL
VÜCUT SENSÖRLERİ	BODY_SENSORS
	USE_BIOMETRIC
	USE_FINGERPRINT

İmza İzinleri

İmza izinleri talep eden uygulama ile bu izinleri tanımlayan uygulamanın aynı sertifika ile imzalanmış olması durumunda verilen izinlerdir. Eğer izni tanımlayan ve talep eden uygulamaların imzalandıkları sertifikalar eşleşirse bu imzalar sistem tarafından kurulum sırasında otomatik olarak verilir. Çizelge 2.6'da örnek olarak imza izinlerinden bazıları ve bu izinlerin açıklaması gösterilmektedir.

Çizelge 2.6. Örnek imza izinleri

İzin Adı
BIND_ACCESSIBILITY_SERVICE
BIND_CARRIER_SERVICES
BIND_DEVICE_ADMIN
BIND_NFC_SERVICE
BIND_TEXT_SERVICE
CLEAR_APP_CACHE
READ_VOICEMAIL
REQUEST_INSTALL_PACKAGES

Android işletim sisteminde çok sayıda izin olmasına ve her sürümde izin sayısı daha da artmasına rağmen izin tabanlı güvenlik politikasının daha çok uygulama geliştirici ya da kullanıcı hataları gibi insan kaynaklı nedenlerle kötücül yazılımlara karşı güvenliği tam olarak sağlayamadığı çeşitli araştırmacılar tarafından yapılan çalışmalarda ortaya konmuştur. Fang ve diğerleri yaptıkları çalışmada izin politikası ile ilgili olarak izinlerin kabaca bölümlenmesi, yeteneksiz izin yöneticileri, yetersiz izin dokümantasyonu, aşırı izin talebi, izin yükseltme saldırısı ve kontrol zamanı kullanım zamanı saldırısı alanlarında sorunlar bulunduğunu belirlemişler ve bu sorunlu alanlarla ilgili olarak literatürde yapılan çalışmaları özetleyerek çeşitli istatistikler sunmuşlardır [8]. Do ve diğerleri yaptıkları çalışmada SMS yoluyla ve duyulamayan ses sinyali yoluyla Android işletim sistemi çalıştıran bir cihazdan bilgi sızdırılabileceğin ortaya koymuşlardır [43]. Bu çalışmada bir saldırgan modeli geliştirmiş, Android cihazlarından verilerin sızdırılabileceği yolları tespit etmiş ve bu yolların sınırlamalarını, gerektirdiği izinleri, kullanım amaçlarını açıklamışlardır. Ayrıca hem mobil cihaz üzerindeki bir dosyanın parçalara ayrılarak SMS yoluyla kullanıcıya hissettirmeden belirli bir başka cihaza aktarılabileceğini hem de insan kulağının duyabileceği 20 Hz ile 20 KHz frekans aralığı dışındaki frekanslarda ses sinyali

göndererek mobil cihazda girilen kullanıcı adı, şifre vb. kısa bilgilerin 6 metreye kadar mesafede bulunan başka bir cihaza gönderilebileceğini testlerle ortaya koymuşlardır. Üstelik ikinci yöntemde hiçbir izin istenmesine gerek olmamıştır, bu nedenle bu yöntemle yapılan bilgi kaçırmanın tespit edilmesi çok zordur.

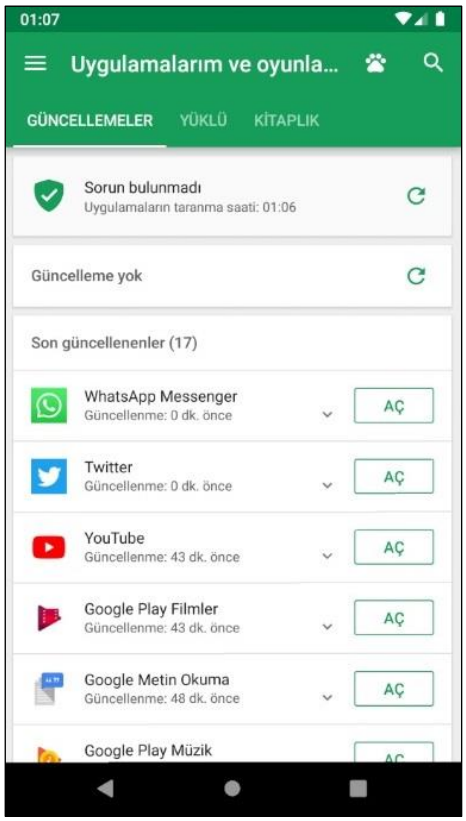
2.4.2. Kullanılan güvenlik sistemleri

Google Play'e yüklenmiş uygulamaların kötücül olup olmadığını tespit etmek ve kötücül uygulamaları mağazadan silmek maksadıyla 2011 yılı içerisinde Google Bouncer isimli bir güvenlik sisteminin devreye alındığı 2012 yılı Şubat ayında Google yöneticisi Hiroshi Lockheimer tarafından duyurulmuştur. Google Bouncer hem yeni yüklenen hem de eskiden yüklenmiş olan uygulamaları ve geliştirici hesaplarını Google'ın bulut altyapısını kullanarak otomatik olarak incelemeye başlamıştır. Google Bouncer'ın devreye alınmasıyla birlikte Google Play'den potansiyel olarak kötücül yazılım indirilme miktarı olarak 2011 yılının ikinci yarısında ilk yarısına göre %40'lık bir düşüş görüldüğü de Google tarafından açıklanmıştır [44].

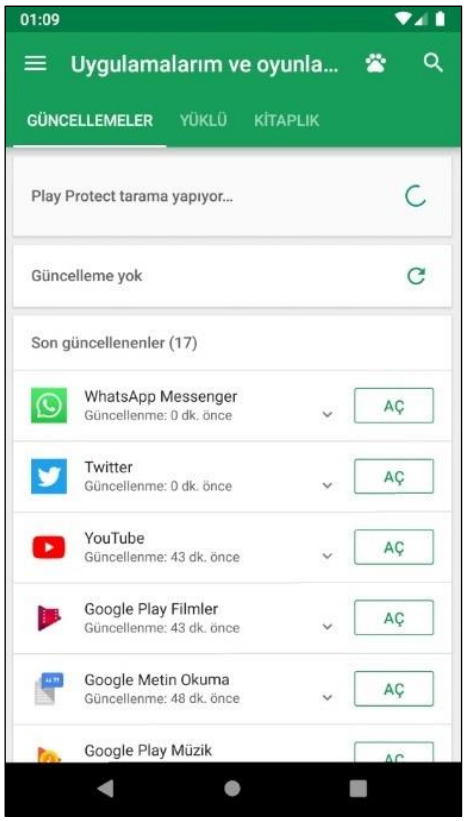
Ancak sistemin devreye alınmasının ardından iki güvenlik araştırmacısının siber saldırı ile Google Bouncer inceleme ortamına sızması sonucunda Google Bouncer'ın sadece dinamik analiz yaptığı, yüklenen uygulamaları beş dakika boyunca incelediği ve internete erişmelerine izin verdiği tespit edilmiştir [45]. Bu tespitler sonucunda TrendMicro güvenlik firması tarafından kötücül kodun gecikmeli olarak çalışması ya da güncellemeyle cihaza indirilmesi gibi çeşitli saldırı teknikleriyle Google Bouncer'ın atlatılabileceği değerlendirilmiştir. [46]. Ayrıca Check Point güvenlik firması tarafından Google Bouncer'ı atlatarak Google Play'e yüklenmiş olan BrainTest isimli bir Android oyun uygulaması içerisinde kötücül bir yazılım bulunduğu açıklanmıştır. Bu uygulamanın iki sürümü Google Play üzerinde yayınlanmış, her iki sürüm de 100 bin ile 500 bin arasında indirilme grubunda yer almıştır, yani toplamda 200 bin ile 1 milyon arasında cihaza bu kötücül uygulama kurulmuştur. Check Point firması uygulamanın Google Bouncer'ı atlatmak amacıyla bu sistem tarafından kullanılan IP adreslerini kontrol etme, zaman bombası, dinamik kod yükleme, yansıtma, kod karmaşıklaştırma gibi teknikler kullandığını ve cihaz üzerine kurulduktan sonra yönetici yetkisi elde etmek amacıyla hak yükseltme saldırısı yaptığını duyurmuştur [47].

Google I/O 2017 konferansında Google Play uygulama mağazasına ve Android cihazlarına yüklenen uygulamaları incelemek ve kötücül yazılımları tespit etmek maksadıyla Google Play Protect isimli yeni bir güvenlik sisteminin tanıtımı yapılmıştır. [48, 49] Bu sistem makine öğrenmesi tekniklerinin de yardımıyla sürekli olarak güncellenmekte ve hem Google Play'e yüklenen hem de Android cihazlarına kurulmuş olan uygulamaları düzenli olarak taramaktadır. Google Play güncellemesi ile birlikte tüm Android cihazlarına kurulan sistem cihazlarda belirli aralıklarla arka planda otomatik olarak çalışmaktadır. Cihaza bilinmeyen bir uygulama yüklenmesi halinde yüklenen uygulama kullanıcı ayarına bağlı olarak sistem tarafından otomatik olarak Google bulutuna yüklenebilmekte ve derin analiz için sıraya alınabilmektedir. 2017 yılı itibariyle sistemin dünya genelinde her gün bir milyardan fazla cihazda elli milyardan fazla uygulamayı taradığı açıklanmıştır. Google Play Protect'in bir bileşeni olarak Cihazımı Bul özelliği de devreye alınmıştır [50].

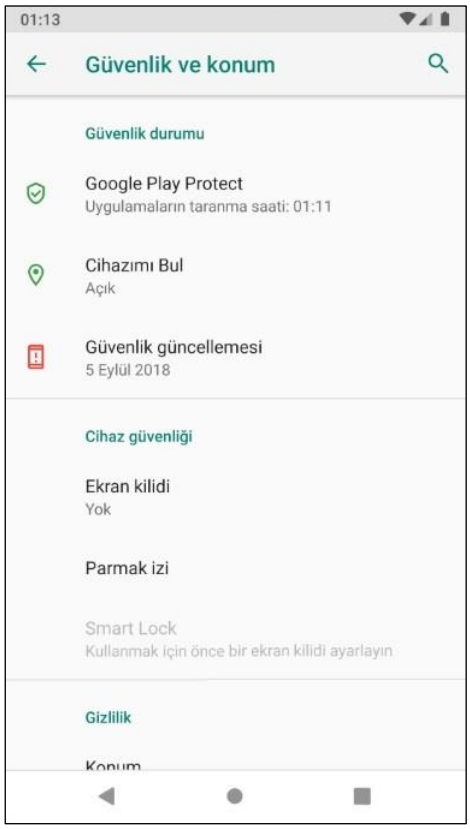
Google Play Protect devreye alındıktan sonra Android cihazında Google Play'in Güncellemeler sayfasında en üste yerleştirilmiştir. Burada cihaz üzerine yüklenmiş olan uygulamaların taranma sonucunu ve son taranma zamanını göstermektedir. Şekil 2.11a'da Google Play'in Güncellemeler sayfası gösterilmektedir. Sistem cihazda belirli aralıklarla otomatik olarak yaparken aynı zamanda kullanıcı tarafından da bu bölümdeki simgeye basılarak bir tarama başlatılabilir. Şekil 2.11b'de kullanıcı tarafından başlatılmış bir tarama gösterilmektedir. Ayrıca Güncellemeler sayfasında yer alan Google Play Protect bölümüne basılarak ya da sistem ayarlarından Güvenlik ve konum seçilerek Google Play Protect menüsüne ulaşılabilir. Bu menüde cihazın güvenlik durumu ve son taranan uygulamalar görülebilir ya da Google Play Protect'in cihaza yüklenmiş olan uygulamaları taramasına veya bilinmeyen uygulamaları Google bulutuna göndermesine ilişkin ayarlar değiştirilebilir. Şekil 2.11c'de sistem ayarlarında bulunan Güvenlik ve konum menüsü, Şekil 2.11d'de Google Play Protect menüsü gösterilmektedir.



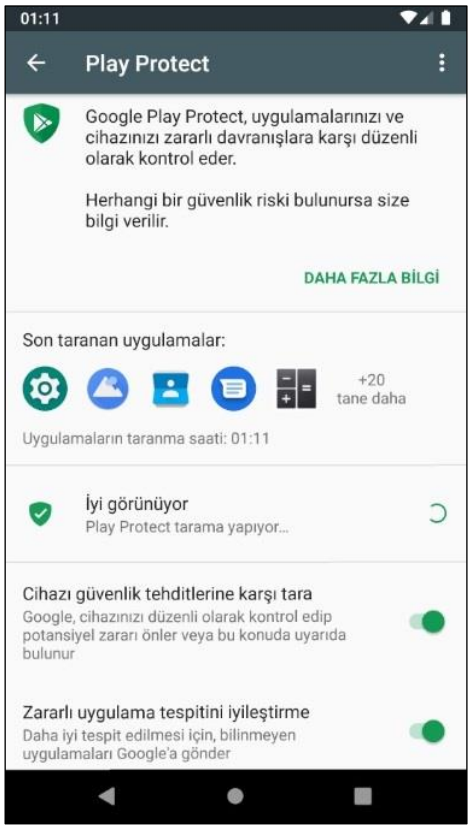
a



b



c



d

Şekil 2.11. a) Güncellemeler sayfası b) Google Play Protect taraması c) Güvenlik ve konum menüsü d) Google Play Protect menüsü

Google ürün yöneticisi Andrew Ahn tarafından yapılan açıklamaya göre yeni makine öğrenmesi modellerinin ve tekniklerinin kullanımı sayesinde 2017 yılında 700 bin uygulama Google Play politikalarını ihlal ettiği için mağazadan kaldırılmıştır. Bu rakam 2016 yılına göre %70 daha fazladır. Ayrıca 2017 yılında 100 bin kötücül uygulama geliştiricinin hesabı da kapatılmıştır. Google Play Protect'in devreye alınması ile birlikte Google Play'e yıllık potansiyel kötücül uygulama yüklenme oranı %50 oranında azalmıştır [51].

Bununla birlikte AV-Comparatives organizasyonu tarafından yapılan mobil güvenlik testlerinde Google Play Protect diğer anti virüs yazılımlarına göre daha düşük bir performans sergilemiştir. AV-Comparatives güvenlik yazılımlarının performanslarını değerlendirmek ve karşılaştırmak maksadıyla düzenli olarak testler yapan ve sonuç raporlarını tüm dünyaya ücretsiz olarak açıklayan Avusturya merkezli bağımsız bir organizasyondur. Avrupa Birliği, Avusturya Tirol bölgesel hükümeti gibi kamu kurumlarının yanı sıra Innsbruck Üniversitesi ve dünya genelindeki çeşitli akademik birimler tarafından desteklenmektedir [52, 53]. Bu organizasyon tarafından 2019 yılı Haziran ayında yapılan testlerde önce birkaç hafta içerisinde 3601 adet kötücül ve çeşitli uygulama mağazalarından 500 adet iyicil yazılım toplanmıştır. Toplanan tüm yazılımlar Avast, AVG, Avira, Bitdefender, F-Secure, G DATA, Kaspersky, McAfee, Securion ve Trend Micro gibi farklı firmaların anti virüs yazılımları ile birlikte Google Play Protect kullanılarak analiz edilmiştir [54]. Analiz sonuçları Çizelge 2.7'de sunulmuştur.

Çizelge 2.7. AV-Comparatives mobil güvenlik testi sonucu [54]

Yazılım Adı	Gerçek Pozitif Oranı	Yanlış Pozitif Sayısı
Trend Micro	100%	0
Avast, AVG, Avira, Bitdefender, F-Secure, G DATA, Kaspersky, McAfee	99.9%	0
Securion	99.4%	2
Google Play Protect	83.2%	28

Test sonuçlarında gerçek pozitif oranı 3601 kötücül yazılımın doğru şekilde kötücül olarak tespit edilme oranını, yanlış pozitif sayısı 500 iyicil yazılımdan yanlış şekilde kötücül olarak tespit edilenlerin sayısını göstermektedir. Test sonuçlarına göre Trend Micro firmasının ürünü en başarılı performansı sergileyerek tüm kötücül ve iyicil yazılımları doğru şekilde tespit etmiştir. Avast, AVG, Avira, Bitdefender, F-Secure, G DATA, Kaspersky ve McAfee firmalarının ürünleri ise iyicil yazılımlarda %100, kötücül yazılımlarda %99,9 oranında

doğru tespit ortaya koymuştur. Testlerde en kötü performansı gösteren Google Play Protect sistemi kötücül yazılımlarda %83,2 oranında doğru tespit yaparken iyicil yazılımlardan 28 adedini yanlışlıkla kötücül olarak belirlemiştir.

Aynı test sonuçları raporunda Amazon Fire OS, LineageOS gibi Android tabanlı işletim sistemlerinde Google uygulamalarının varsayılan olarak bulunmadığı ve dolayısıyla Google Play Protect koruması olmadığı da belirtilmiştir. Ayrıca Avrupa ve Amerika'da piyasaya hâkim olan Google Play'de düzenli ve sıkı bir inceleme olması nedeniyle buradan yanlışlıkla kötücül yazılım indirme ihtimalinin nispeten düşük olduğu ancak başta Çin olmak üzere çoğu Asya ülkesinde piyasada çok fazla üçüncü taraf uygulama mağazası olması nedeniyle buralarda kötücül yazılım bulaşma ihtimalinin çok daha yüksek olduğu iddia edilmiştir. Rapora göre Çin'de 787 milyondan fazla kişi mobil cihaz kullanmakta ve bunların yaklaşık %75'i işletim sistemi olarak Android kullanmaktadır. Çin'de en çok kullanılan Android uygulama mağazaları Şekil 2.12'de gösterilmiştir.

	Uygulama Mağazası	Pazar Payı (%)	Son ay en az bir kez kullanan kişilerin Sayısı	Son ay en az bir kez kullanan kişilerin Değişimi (%)	Son ay ortalama günlük kullanıcı Sayısı	Son ay ortalama günlük kullanıcı Değişimi (%)
1	Tencent My App	26.03	258,116,000	-2.80	59,920,000	+0.50
2	Huawei App Market	12.76	126,548,000	+0.90	17,419,000	+4.80
3	Oppo Software Store	12.35	122,458,000	-2.70	23,414,000	+5.20
4	360 Mobile Assistant	9.71	96,244,000	-1.70	42,768,000	-0.10
5	Baidu Mobile Assistant	9.15	90,717,000	+6.70	12,144,000	-0.30
6	MIUI App Store	8.71	86,352,000	+0.10	16,407,000	+0.50
7	VIVO App Store	7.34	72,821,000	+2.50	16,020,000	+1.60
8	PP Assistant	2.57	25,446,000	+1.00	4,784,000	+0.50
9	China Mobile MM Store	2.02	20,031,000	+3.80	1,645,000	-3.60
10	Anzhi Market	1.82	18,074,000	+0.40	3,470,000	-1.70

Şekil 2.12. Çin'de en çok kullanılan Android uygulama mağazaları [55]

Çin'de Tencent My App uygulama mağazası açık bir farkla en fazla kullanılan mağazayken Huawei App Market, Oppo Software Store, 360 Mobile Assistant mağazaları onu takip etmektedir. Çin'de Google Play'in ve pek çok Google hizmetinin yasaklanmış olması nedeniyle Google Play mağazası bu listeye girememiştir.

ESET firmasının kötücül yazılım araştırmacılarından Lukas Stefanko'nun Google Play'deki kötücül yazılımlara ilişkin 2019 yılı Temmuz ayı araştırmasına göre Google Play'den Temmuz ayında toplamda 205 kötücül uygulama toplamda 32 milyondan fazla kez indirilmiştir [56]. Ayrıca 2019 yılı Haziran ayında Sydney Üniversitesi'nin de katıldığı bir kötücül yazılım araştırma çalışmasında Google Play üzerindeki en popüler 10 bin uygulamanın sahte kopyalarını tespit etmek maksadıyla ikon tasarımlarını ve açıklamaları incelemek üzere sinir ağları kullanılmıştır. Toplamda 1,2 milyon uygulama incelenmiş ve en popüler 10 bin uygulamaya çok büyük benzerlik gösteren 49608 uygulama arasından 2040 tanesinin kötücül yazılım içerdiği bulunmuştur. Araştırmada aynı zamanda 1565 sahte uygulamanın orijinal uygulamaya göre en az beş adet ilave tehlikeli izin talebinde bulunduğu ve 1407 sahte uygulamanın en az beş adet üçüncü taraf reklam kütüphanesi içerdiği de bulunmuştur [57].

Kötücül yazılımların Google Play mağazasına yüklenmesini ve Android cihazlarına bulaşmasını engellemek maksadıyla Google tarafından güvenlik mekanizması sürekli olarak geliştirilmektedir. Ancak yapılan çalışmalarda Google Play Protect sisteminin kötücül yazılımları tespit etme ve engelleme konusunda yeterli performansı sunamadığı, Google Play mağazasında hâlâ binlerce kötücül yazılım bulunduğu ortaya konmuştur. Ayrıca Android tabanlı bazı mobil işletim sistemlerinde Google hizmetleri ve Google Play Protect servisi hazır kurulu olarak gelmemektedir. Bunların dışında Google Play dışında pek çok uygulama mağazası da bulunmakta ve bunlar özellikle Avrupa ve Amerika dışında çok yoğun olarak kullanılmaktadır. Bu nedenle Android işletim sistemine yönelik olarak geliştirilen kötücül yazılımları tespit eden ya da bu kötücül yazılımların etkilerini azaltan, hassas verilere erişmelerini ve sızdırmalarını engelleyen güvenlik çözümlerine hâlâ çok büyük ihtiyaç vardır.

3. LİTERATÜRDE YER ALAN ÇALIŞMALAR

Geliştirilmiş olan yazılımlara ilişkin detaylı bilgiler toplayabilmek ve kötücül yazılımları tespit edebilmek için statik analiz, dinamik analiz ve ikisinin birleşimi olan hibrit analiz gibi çeşitli analiz yöntemleri kullanılmaktadır. Bu bölümde Android işletim sistemindeki uygulamalarla ilgili olarak yapılan çalışmalar kullandıkları analiz yöntemine göre alt bölümlerde incelenecektir.

3.1. Statik Analiz

Statik analiz yönteminde incelenen bir APK uygulamasının AndroidManifest.xml dosyası veya kodu okunarak uygulama çalıştırılmadan çeşitli bilgiler elde edilmeye çalışılmaktadır. Bu bölümde statik analiz yöntemini kullanan çalışmalar incelenecektir.

Android Permissions Demystified

Felt ve diğerleri yaptıkları çalışmada derlenmiş Android uygulamalarında talep edilen aşırı izinleri belirlemek üzere Stowaway adlı bir araç geliştirmişlerdir [58]. Geliştirilen araç, statik analiz tekniklerini kullanarak verilen bir uygulamanın yaptığı API çağrılarını bulmakta ve ardından bu çağrıları izinlerle eşleştirmektedir. Her bir API çağrısı için gereken izin ya da izinleri bulmak için izin haritası kullanılmıştır. İzin haritasını oluşturmak amacıyla Android API içerisinde bulunan otomatik test araçlarından faydalanılmış ve toplamda 1259 API çağrısı izinlerle eşleştirilmiştir. Bu yolla uygulamanın gerektirdiği en büyük izin kümesi belirlenmektedir.

Çalışma kapsamında kullanmak maksadıyla 2010 yılının Ekim ayında en popüler 100 ücretli uygulama, en popüler 764 ücretsiz uygulama ve Android Market'e yeni eklenmiş 100 ücretsiz uygulama olmak üzere toplam 964 Android uygulamasından oluşan bir veri kümesi oluşturmuşlardır. Bu uygulamalardan rastgele seçilen 24 adedini eğitim ve test için, kalan 940 adedini ise analiz için kullanmışlardır. Yapılan analizler sonucunda uygulamaların yaklaşık üçte birinin aşırı izin talebinde bulunduğu tespit edilmiştir. Aşırı izin talep eden uygulamaların %56'sı tek bir gereksiz izin talep ederken sadece %6'sı dörtten fazla gereksiz izin talebinde bulunmaktadır. Bu da uygulama başına gereksiz izin talebinin çok fazla

olmadığını göstermektedir. Yazarlar buradan yola çıkarak uygulama geliştiricilerin en az izin ilkesine uymaya çalıştıklarını ancak API dokümantasyonundaki yetersizlik ve hatalar nedeniyle tam olarak yapamadıklarını değerlendirmişlerdir. Yapılan analizlerde en çok talep edilen beş gereksiz izin olarak ACCESS_NETWORK_STATE, READ_PHONE_ STATE, ACCESS_WIFI_STATE, WRITE_EXTERNAL_STORAGE ve CALL_PHONE izinleri bulunmuştur.

PScout: Analyzing the Android Permission Specification

Au ve diğerleri yaptıkları çalışmada Android izin sistemine ilişkin dokümantasyonun eksik olduğunu iddia etmişler ve statik analiz teknikleriyle Android kaynak kodunu inceleyerek izin gereksinimlerini çıkaran PScout aracını geliştirmişlerdir [59]. Bu araç Android işletim sisteminin sürümünden bağımsız olarak çalışmakta ve Android işletim sisteminin kodunda geçen tüm API çağrılarının gerektirdiği izinleri bulmaktadır. Aracın çalışması sonucunda API çağrıları ile izinler arasında bir eşleşme tablosu oluşturulmaktadır.

Geliştirilen aracın etkinliğini değerlendirmek amacıyla Android 2.2 ile Android 4.0 arasındaki dört adet işletim sürümü analiz edilmiştir. Yapılan analiz sonucunda API çağrıları ile izinler arasında 17 218 adet eşleşme bulunmuştur. Ayrıca API çağrılarının %80'den fazlasının sadece bir izin gerektirdiği ve izinlerin %75'inin yirmiden daha düşük sayıda API çağrısı tarafından kontrol edildiği de ortaya çıkarılmıştır. Bunun sonucunda API çağrıları ile izinlerin çok sıkı bir şekilde bağlı olmadığı sonucuna varılmıştır.

Ardından Android Market'ten indirdikleri 1260 adet uygulamayı Android 2.2 sürümü üzerinde incelemişlerdir. İncelemede uygulamalar tarafından talep edilen izinler ile PScout tarafından bulunan izinler karşılaştırılmış ve 1260 uygulamadan 543 tanesinin fazladan bir adet izin talep ettiği bulunmuştur. Uygulamaların %53'ü bir adet fazla izin talep ederken sadece %5'i fazladan dört veya daha çok izin talep etmiştir.

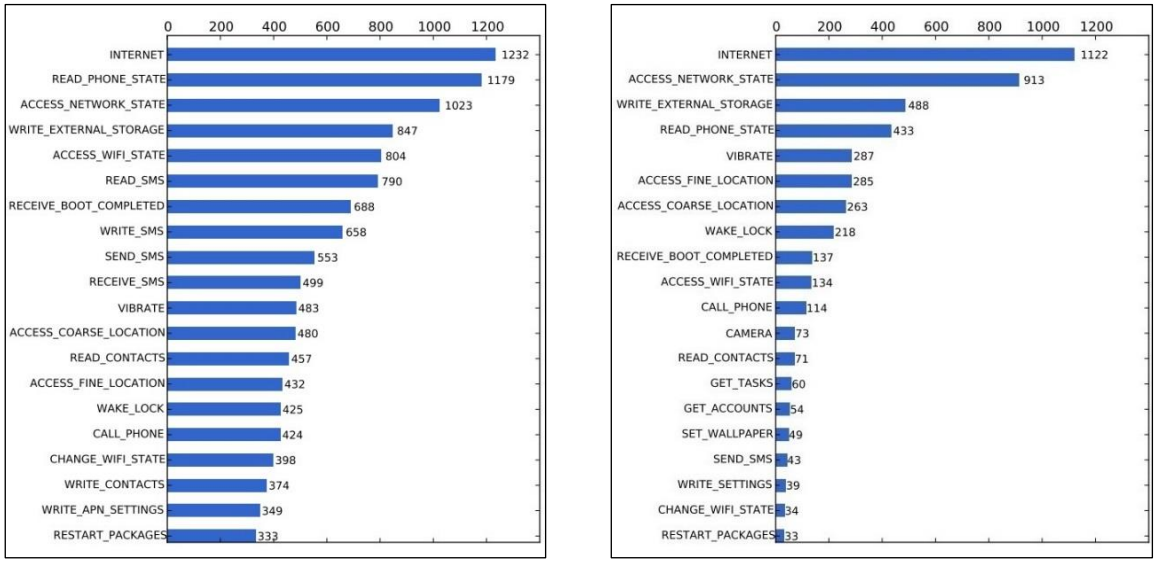
Dissecting Android Malware: Characterization and Evolution

Zhou ve Jiang yaptıkları çalışmada Android kötücül yazılımlarını sistematik olarak karakterize etmeyi amaçlamışlardır. Bunun için çeşitli Android uygulama marketlerinden 2010 yılı Ağustos ayı ile 2011 yılı Ekim ayı arasında 49 farklı aileden toplam 1260 adet

kötücül yazılım toplamışlardır. Ardından bunları kurulum yöntemine, etkinleşme mekanizmasına, taşıdıkları kötücül yüke ve yaptıkları işlemlere göre sınıflandırmışlardır. Toplanan kötücül yazılımlar Android Malware Genome Project adıyla araştırmacılara kötücül yazılım veri kümesi olarak sunulmaktadır.

Yapılan çalışmalar soncunda veri kümesindeki 1260 kötücül yazılımdan 1083 adedinin (%86,0) iyicil uygulamaların içerisine kötücül kod eklendikten sonra yeniden paketlenerek oluşturulduğu bulunmuştur. Ayrıca kötücül yazılımların %36,7'sinin yönetici seviyesi açıklıkları kullandığı, %93'ünün telefonu uzaktan ağ üzerinden ya da SMS ile kontrol edilebilen bir köle telefona çevirdiği tespit edilmiştir. Bunların dışında 49 aileden 28 adedindeki uygulamaların (571 uygulama, %45,3) arka planda kullanıcı farkında olmadan ücretli numaralara mesaj atabildiği ya da arama yapabildiği bulunmuştur. Son olarak 27 ailedeki uygulamaların (644 uygulama, %51,1) kullanıcının hesap bilgileri ve mesajları gibi kişisel bilgilerini topladığı ortaya çıkarılmıştır.

Kötücül yazılımlar ile iyicil yazılımlar arasında izin talebi karşılaştırması yapabilmek için 2011 yılının Ekim ayının ilk haftasında resmi Android uygulama mağazasından en popüler ücretsiz 1260 adet iyicil uygulama indirmişlerdir. 1260 adet kötücül ve iyicil yazılım tarafından en çok talep edilen izinlerin karşılaştırması Şekil 3.1a ve Şekil 3.1b'de gösterilmektedir [60].

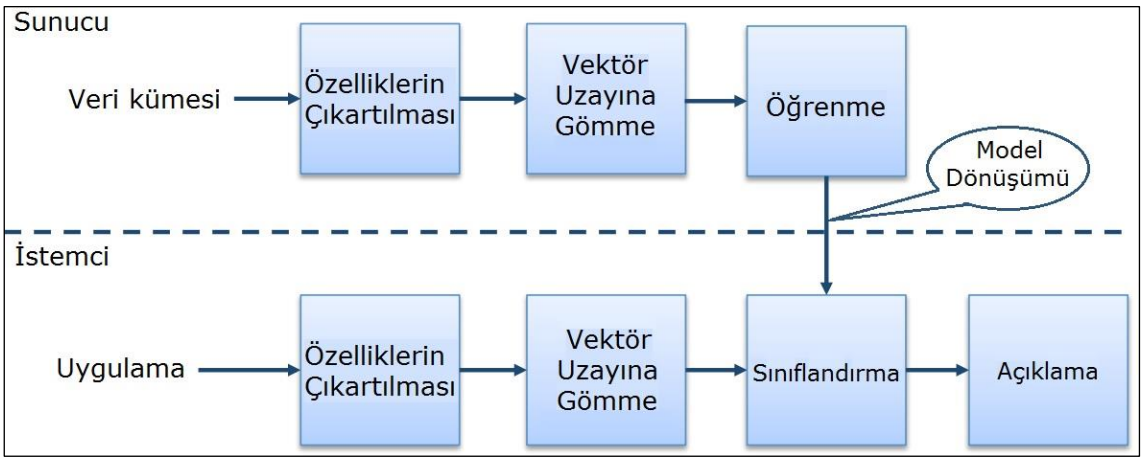


a b

Şekil 3.1. a) 1260 kötücül yazılımın en çok talep ettiği 20 izin b) 1260 iyicil yazılımın en çok talep ettiği 20 izin [60]

Drebin: Effective and Explainable Detection of Android Malware in Your Pocket

Arp ve diğerleri yaptıkları çalışmada imza tabanlı kötücül uygulama tespit sistemlerinin bilinmeyen kötücül uygulamaları tespit edememe sorununa çözüm bulmaya çalışmışlar ve bu amaçla mobil cihaz üzerinde statik analiz yapan Drebin isimli bir yazılım geliştirmişlerdir [61]. Drebin yazılımının çalışma mantığı Şekil 3.2'de sunulmuştur.



Şekil 3.2. Drebin yazılımının çalışma mantığı [61]

Drebin yazılımı özelliklerin çıkartılması bölümünde AndroidManifest.xml dosyasından uygulama bileşenlerini, filtrelenmiş amaçları, donanım bileşenlerini ve talep edilen izinleri, uygulama kodundan da korunan API çağrılarını, kullanılan izinleri, şüpheli API çağrılarını ve ağ adreslerini çıkartmaktadır. Ardından çıkartılan tüm özellikler bir vektör üzerinde 1'e eşitlenerek gösterilmiş ve benzer özelliklere sahip uygulamalar uzayda belirli bir bölgede toplanmıştır. Drebin vektör ve uygulama uzayı gösterimi Şekil 3.3'te sunulmuştur.

Şekil 3.3. Drebin vektör ve uygulama uzayı gösterimi [61]

Öğrenme amacıyla doğrusal destek vektör makineleri kullanılmış ve bu yöntemle iyicil ve kötücül uygulamalar arasında en büyük aralıkla bir hiperdüzlem oluşturulmaktadır. Daha sonra ayrık bir sistemde oluşturulan öğrenilmiş tespit modeli akıllı telefona aktarılmaktadır. Geliştirilen yazılımı test etmek maksadıyla Mobile-Sandbox projesiyle toplanan 123 453 iyicil ve 5560 kötücül yazılım kullanılmıştır. Kötücül yazılımlar akademik çalışmaları desteklemek amacıyla Drebin veri kümesi adıyla araştırmacıların erişimine de açılmıştır. Toplanan uygulamaların %66'sı eğitim, %33'ü test maksadıyla kullanılmıştır. Yapılan testler sonucunda Drebin yazılımının %1 yanlış pozitif oranıyla kötücül uygulamaları %94 oranında tespit edebildiği görülmüştür.

Mining Permission Patterns for Contrasting Clean and Malicious Android Applications

Moonsamy ve diğerleri yaptıkları çalışmada talep edilen veya kod içerisinde kullanılan izinlerden izin modelleri oluşturmak ve bu modelleri kullanmak suretiyle kötücül yazılımları tespit etmeye çalışmışlardır [62]. İzin modellerini belirlemek amacıyla kullanılabilecek yöntemler konusunda mevcut literatürde eksiklikler bulunduğunu değerlendirerek bu konuyu araştırmışlardır. Öncelikle iyicil ve kötücül uygulama veri kümelerindeki uygulamalar tarafından en çok talep edilen izinleri ve en çok kullanılan izinler belirlenmiştir. Yaptıkları incelemeye göre iyicil ve kötücül uygulamalar tarafından en çok talep edilen izinler Şekil 3.4'te gösterilmiştir.

İyicil ve kötücül uygulamalar tarafından en çok talep edilen 20 izin

İyicil uygulamalar		Kötücül uygulamalar	
Talep edilen izin	**Frekans**	**Talep edilen izin**	**Frekans**
INTERNET	1121 (91.36%)	INTERNET	1199 (97.72%)
ACCESS_NETWORK_STATE	663 (54.03%)	ACCESS_COARSE_LOCATION	1146 (93.40%)
READ_PHONE_STATE	391 (31.87%)	VIBRATE	994 (81.01%)
WRITE_EXTERNAL_STORAGE	362 (29.50%)	WRITE_EXTERNAL_STORAGE	823 (67.07%)
ACCESS_COARSE_LOCATION	236 (19.23%)	READ_SMS	779 (63.49%)
VIBRATE	210 (17.11%)	WRITE_SMS	762 (62.10%)
WAKE_LOCK	188 (15.32%)	READ_CONTACTS	680 (55.42%)
ACCESS_FINE_LOCATION	162 (13.20%)	BLUETOOTH	633 (51.59%)
GET_TASKS	125 (10.19%)	WRITE_CONTACTS	542 (44.17%)
SET_WALLPAPER	102 (8.31%)	DISABLE_KEYGUARD	491 (40.02%)
ACCESS_WIFI_STATE	64 (5.22%)	WAKE_LOCK	471 (38.39%)
RECEIVE_BOOT_COMPLETED	60 (4.89%)	RECORD_AUDIO	461 (37.57%)
READ_CONTACTS	58 (4.73%)	ACCESS_FINE_LOCATION	446 (36.35%)
WRITE_SETTINGS	45 (3.67%)	ACCESS_NETWORK_STATE	416 (33.90%)
CAMERA	43 (3.50%)	READ_PHONE_STATE	414 (33.74%)
CALL_PHONE	42 (3.42%)	SET_ORIENTATION	413 (33.66%)
SEND_SMS	34 (2.77%)	CHANGE_WIFI_STATE	384 (31.30%)
RESTART_PACKAGES	32 (2.61%)	READ_LOGS	361 (29.42%)
RECEIVE_SMS	31 (2.53%)	BLUETOOTH_ADMIN	342 (27.87%)
RECORD_AUDIO	27 (2.20%)	RECEIVE_BOOT_COMPLETED	325 (26.49%)

Şekil 3.4. İyicil ve kötücül uygulamalar tarafından en çok talep edilen 20 izin [62]

Belirledikleri izinlerin karşılaştırma sonucuna göre kötücül uygulamalar toplamda 14758 adet izin isterken iyicil uygulamalar 4470 adet izin istemiştir. Toplamda sadece iyicil uygulamaların istediği 33 izin, sadece kötücül uygulamaların istediği 20 izin, ortak olarak istenen 70 izin ve hiç istenmeyen 5 izin tespit edilmiştir. Hem iyicil hem de kötücül uygulamalar tarafından en çok istenen izinler INTERNET, ACCESS_COARSE_ LOCATION, WRITE_EXTERNAL_STORAGE ve VIBRATE izinleridir.

Moonsamy ve diğerlerinin yaptıkları araştırmaya göre iyicil ve kötücül uygulamaların en çok kullandıkları izinler Şekil 3.5'te gösterilmiştir.

İyicil ve kötücül uygulamalar tarafından en çok kullanılan 20 izin

İyicil uygulamalar		Kötücül uygulamalar	
Kullanılan izin	**Frekans**	**Kullanılan izin**	**Frekans**
INTERNET	1029 (83.86%)	INTERNET	1161 (94.62%)
WAKE_LOCK	816 (66.50%)	ACCESS_COARSE_LOCATION	1125 (91.69%)
ACCESS_NETWORK_STATE	738 (60.15%)	VIBRATE	954 (77.75%)
VIBRATE	608 (49.55%)	WAKE_LOCK	826 (67.32%)
READ_PHONE_STATE	457 (37.25%)	ACCESS_WIFI_STATE	584 (47.60%)
ACCESS_COARSE_LOCATION	372 (30.32%)	ACCESS_NETWORK_STATE	519 (42.30%)
SET_WALLPAPER	126 (10.27%)	READ_SMS	473 (38.55%)
ACCESS_FINE_LOCATION	116 (9.45%)	WRITE_CONTACTS	426 (34.72%)
GET_ACCOUNTS	98 (7.99%)	READ_PHONE_STATE	354 (28.85%)
ACCESS_WIFI_STATE	85 (6.93%)	RECORD_AUDIO	319 (26.00%)
READ_SMS	82 (6.68%)	SET_WALLPAPER	297 (24.21%)
RESTART_PACKAGES	65 (5.30%)	ACCESS_FINE_LOCATION	199 (16.22%)
GET_TASKS	61 (4.97%)	GET_ACCOUNTS	178 (14.51%)
CHANGE_CONFIGURATION	55 (4.48%)	GET_TASKS	124 (10.11%)
RECEIVE_SMS	37 (3.02%)	RECEIVE_BOOT_COMPLETED	111 (9.05%)
FLASHLIGHT	37 (3.02%)	ACCESS_CACHE_FILESYSTEM	101 (8.23%)
WRITE_CONTACTS	34 (2.77%)	WRTIE_OWNER_DATA	59 (4.81%)
RECEIVE_BOOT_COMPLETED	23 (1.87%)	CHANGE_CONFIGURATION	52 (4.24%)
WRTIE_OWNER_DATA	12 (0.98%)	READ_HISTORY_BOOKMARKS	49 (3.99%)
WRITE_SETTINGS	10 (0.81%)	EXPAND_STATUS_BAR	41 (3.34%)

Şekil 3.5. İyicil ve kötücül uygulamalar tarafından en çok kullanılan 20 izin [62]

Toplamda sadece iyicil uygulamaların kullandığı 9 izin, sadece kötücül uygulamaların kullandığı 4 izin, ortak olarak kullanılan 28 izin ve hiç kullanılmayan 87 izin tespit edilmiştir.

Doğrudan frekans sayımı gibi klasik istatistiksel yöntemler her bir veri kümesinde en popüler olan tekil izinleri belirlemek için faydalıdır. Ancak tekil izinler yerine izin kombinasyonlarını karşılaştırmak daha karmaşık ve zaman alıcı hale gelmektedir. Bu nedenle yazarlar Android uygulamalarının izin modellerini belirlemek amacıyla literatürde kullanılmakta olan mevcut istatistiksel yöntemleri model madenciliği tekniklerini uygulayarak geliştirmişlerdir. Ardından kötücül ve iyicil uygulama veri kümeleri arasındaki

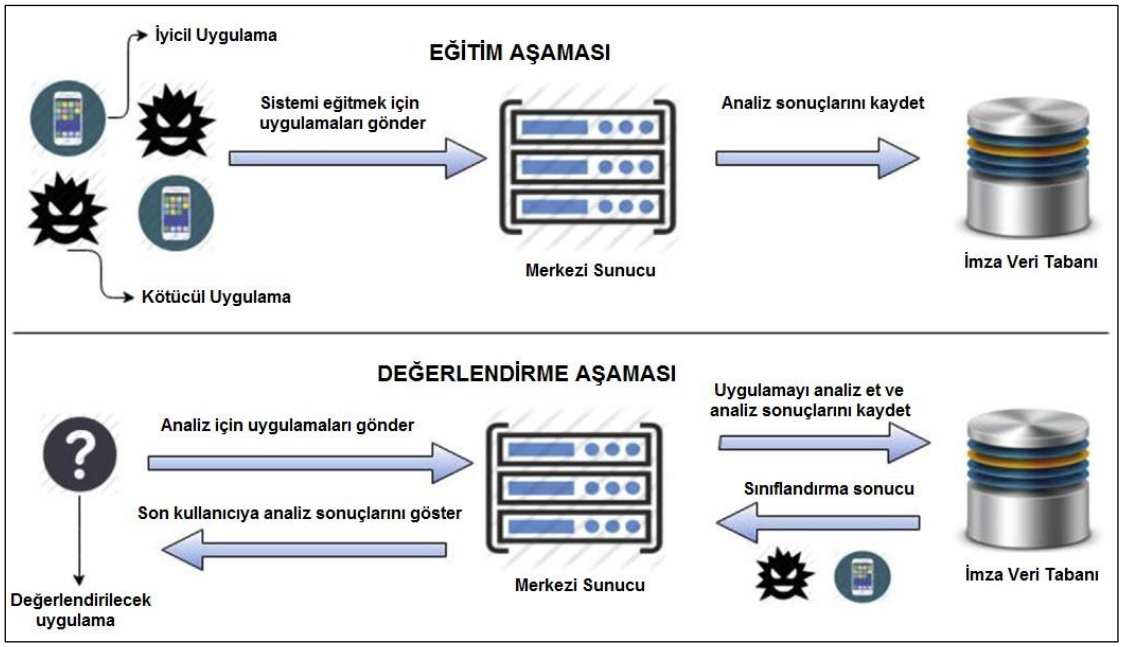
benzerlikleri ve farklılıkları daha iyi gözlemlemek amacıyla geliştirdikleri yöntemi bu iki veri kümesi üzerinde uygulamışlardır. İkili kümeleme yöntemini kullanarak talep edilen ve kullanılan izinler ve uygulamalar arasındaki ilişkileri görsel grafikler olarak ortaya çıkarmışlardır. Daha sonra bu grafiklerde belirli bölgelerde toplanan izin modelleri kullanılarak iyicil ve kötücül uygulamaların ayrılabileceğini değerlendirmişlerdir. Bu amaçla her bir veri kümesinde hangi modellerin daha fazla öne çıktığını belirlemek için karşıt izin modeli madenciliği yöntemini kullanmışlar ve en fazla dört elemana kadar izin grupları oluşturmuşlardır. Daha sonra iyicil uygulamalar ile kötücül uygulamalar arasında en fazla ayrım gösteren izin gruplarını belirlemişler ve bu yolla kötücül uygulamalar ile iyicil uygulamaları ayırmayı hedeflemişlerdir.

APK Auditor: Permission-Based Android Malware Detection System

Kabakuş ve diğerleri yaptıkları çalışmada sunucu-istemci mimarisinde çalışan APK Auditor adlı bir sistem geliştirmişlerdir [63]. Geliştirdikleri sistem bünyesinde Android uygulamalarını inceleyen bir merkezi analiz sunucusu, inceleme sonuçlarını depolayan bir veri tabanı ve son kullanıcılar tarafından uygulama analiz taleplerinin gönderildiği bir Android uygulaması bulunmaktadır. Merkezi analiz sunucusu son kullanıcılar tarafından analiz edilmesi talep edilen bir uygulamanın son sürümünü Google Play mağazasından indirip analiz etmektedir. Analiz işleminde, öncelikle uygulamanın AndroidManifest.xml dosyası içerisinde talep ettiği izinler bulunmaktadır. Ardından bu izinlerden her birinin kötücül yazılımlarda bulunma oranı veri tabanından sorgulanarak tespit edilmekte ve bu orana göre her bir izin için ayrı ayrı izin kötücül yazılım puanı hesaplanmaktadır. Daha sonra uygulamanın talep ettiği tüm izinler için ayrı ayrı hesaplanan izin kötücül yazılım puanları toplanarak uygulama kötücül yazılım puanı bulunmaktadır. Bu puan APK Auditor tarafından belirlenen bir eşik değerin üzerindeyse uygulamanın kötücül olduğu değerlendirilmektedir. Geliştirilen sistemde iyicil ve kötücül uygulamaları belirlenmek için kullanılan eşik değer eğitim aşamasında lojistik regresyon yöntemi kullanılarak belirlenmiştir. Merkezi analiz sunucusu sistemi geliştirmek amacıyla ayrıca otomatik olarak Google Play uygulama mağazasındaki en popüler 100 uygulamayı indirip analiz etmektedir.

Geliştirilen sistemin eğitim ve değerlendirme aşamalarında contagio mobile, Drebin ve Android Malware Genome Project veri kümelerinden oluşan toplam 6909 kötücül uygulama ve Google Play mağazasından indirilen 1853 iyicil uygulama kullanılmıştır. Literatürde

eğitim ve değerlendirme aşamalarında kullanılacak uygulamaların %70'e %30 olacak şekilde ayrılmasının tavsiye edildiği belirtilerek 6133 uygulama eğitim aşamasında, 2629 uygulama ise değerlendirme aşamasında kullanılmıştır. APK Auditor 2629 uygulamadan 2321 adedini doğru şekilde sınıflandırmış ve %88,28'lik bir doğruluk oranına ulaşmıştır. Geliştirilen sistemin mimari yapısı Şekil 3.6'da gösterilmiştir.

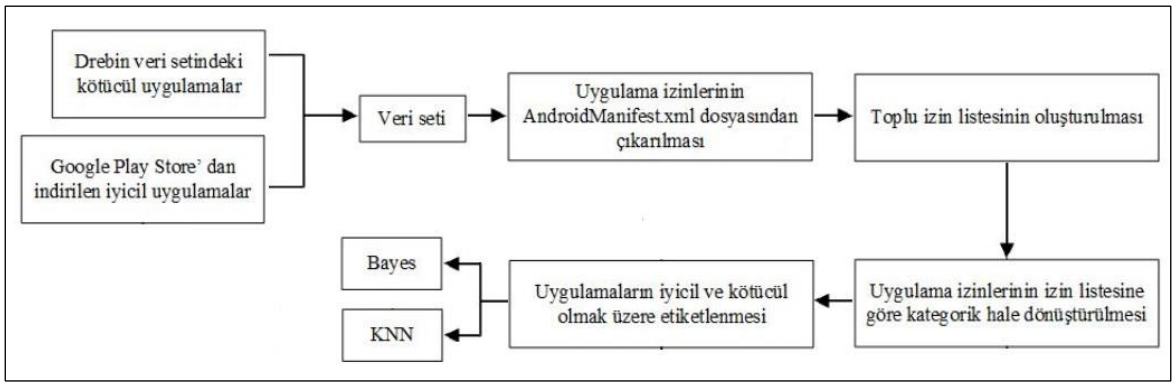


Şekil 3.6. APK Auditor sisteminin mimari yapısı [63]

Android Kötücül Yazılımlar için İzin Tabanlı Tespit Sistemi

Utku ve Doğru yaptıkları çalışmada analiz edilecek uygulamanın manifest izinlerini çıkarmış, bu izinleri kategorilendirmiş ve ardından Naive Bayes ve KNN makine öğrenmesi algoritmaları kullanılarak iyicil ve kötücül yazılımları ayrıştırmışlardır [64]. Geliştirilen sistemin çalışma mantığı Şekil 3.7'de gösterilmiştir.

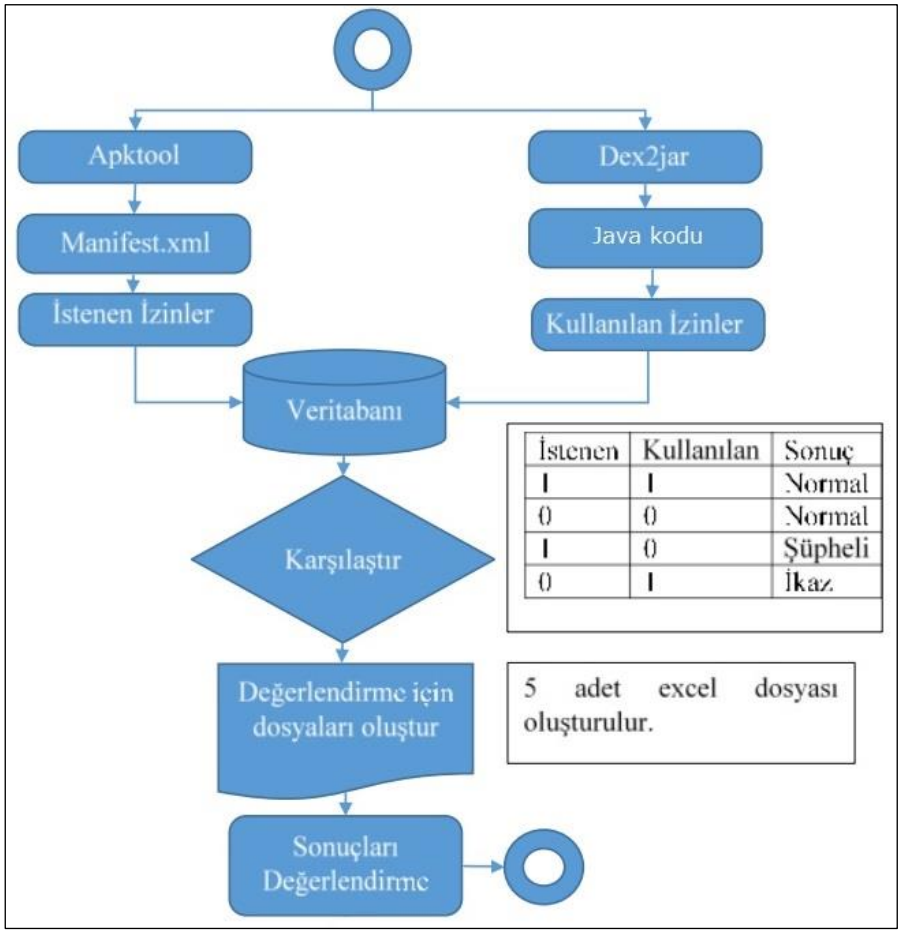
Geliştirdikleri sistemi değerlendirmek amacıyla Drebin veri kümesindeki 5555 kötücül uygulama ile Google Play uygulama mağazasından indirilmiş 1339 iyicil uygulamayı yaklaşık %80'i eğitim, %20'si test için olacak şekilde kullanmışlardır. Yapılan testlerin sonucunda Naive Bayes algoritması kullanıldığında kötücül uygulamalar %97,29, iyicil uygulamalar %77,72 oranında, KNN algoritması kullanıldığında kötücül uygulamalar %97,74, iyicil uygulamalar %48,90 oranında doğru tespit edilmiştir.



Şekil 3.7. Utku ve Doğru'nun geliştirdiği sistemin akış şeması [64]

Android Mobil Uygulamalar için İzin Karşılaştırma Tabanlı Kötücül Yazılım Tespiti

Arslan ve diğerleri yaptıkları çalışmada uygulamaların talep ettikleri izinler ile kod içerisinde kullandıkları izinler arasındaki farklılığa dayanarak kötücül uygulamaları tespit eden bir sistem geliştirmişlerdir [65]. Sistemin akış şeması Şekil 3.8'de gösterilmiştir.

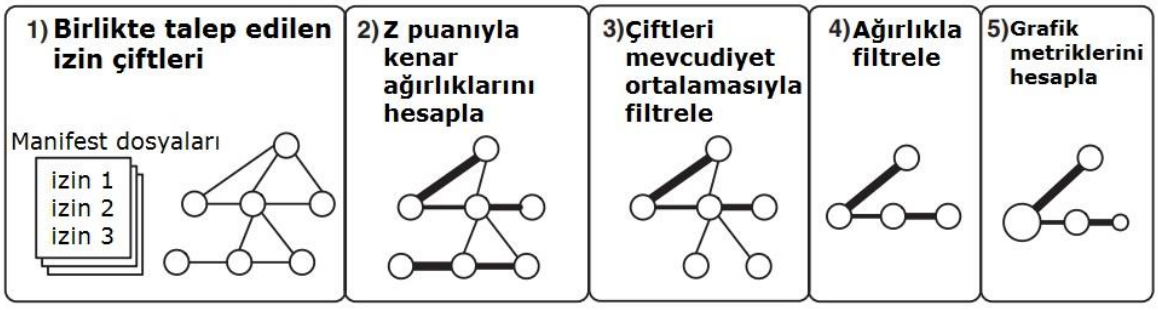


Şekil 3.8. Arslan ve diğerlerinin geliştirdiği sistemin akış şeması [65]

Geliştirilen sistem bir yandan Apktool aracı ile uygulama dosyalarını çözerek AndroidManifest.xml dosyalarını bulmakta ve talep ettikleri izinleri çıkarmakta, diğer yandan Dex2Jar aracı ile uygulama dosyalarını çözerek Java kodlarını ve kod içerisinde kullandıkları izinleri bulmaktadır. Ardından talep edilen izinler ile kod içerisinde kullanılan izinler karşılaştırılmakta ve talep edilen ancak kullanılmayan izinler şüpheli durum, talep edilmeyen ancak kullanılan izinler ise ikaz durumu olarak belirlenmektedir. Daha sonra uygulamaların şüphe değerleri hesaplanarak iyicil ile kötücül uygulamalar bu değere göre ayrıştırılmaktadır. Sistemin değerlendirilmesi maksadıyla 50 kötücül ve 25 iyicil uygulama kullanılmıştır. Yapılan testler sonucunda sistemin kötücül uygulamaları %80 başarı oranıyla tespit edebildiği ve uç değerlerin elenmesi halinde başarı oranının %97,62'ye kadar çıkabildiği bulunmuştur.

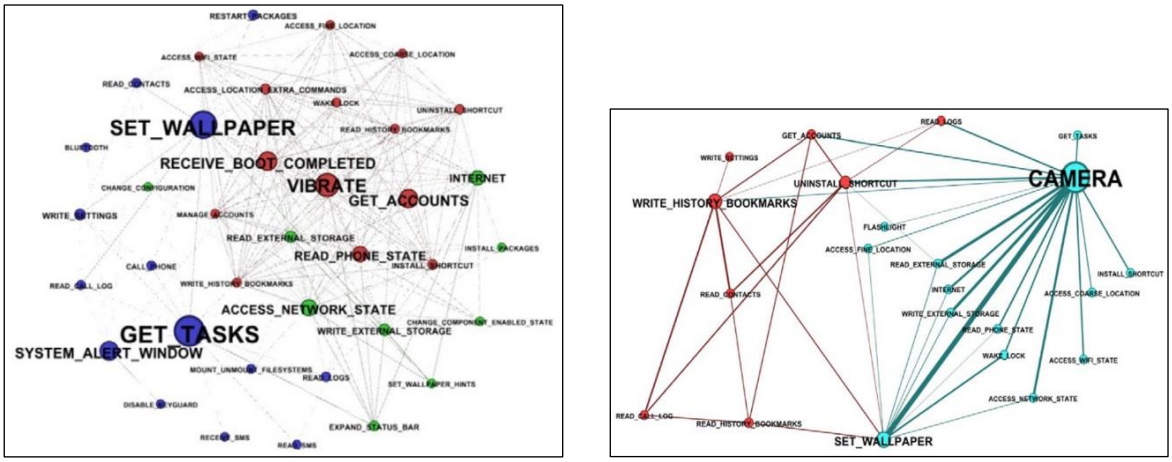
Android Application Classification and Anomaly Detection With Graph-Based Permission Patterns

Sokolova ve diğerleri yaptıkları çalışmada aynı kategori içerisindeki uygualmaların benzer işlevlere sahip olacağı ve bu nedenle benzer izinler talep edeceğini varsaymışlardır. Buna bağlı olarak beklenen izin taleplerini değerlendirerek her bir uygulama kategorisi için normal davranışları karakterize eden bir metodoloji önermişlerdir [66]. Birlikte talep edilen izinler bir grafik olarak modellenmiş ve grafik modelleri ile merkezi izinler grafik analiz metrikleri kullanılarak elde edilmiştir. Beş adımlı model oluşturma metodolojisi Şekil 3.9'da gösterilmiştir. Elde edilen modeller kategorileri en iyi temsil eden grafik metriklerini seçebilmek maksadıyla uygulamaların kategorilere sınıflandırılma performansı ile değerlendirilmiştir.



Şekil 3.9. Sokolova ve diğerlerinin beş adımlı model oluşturma metodolojisi [66]

Önerilen metodolojiyle oluşturulan kategori izin modellerinden örnek olarak kişiselleştirme kategorisinin izin modeli Şekil 3.10a'da ve fotoğrafçılık kategorisinin izin modeli Şekil 3.10b'de gösterilmiştir.



a b

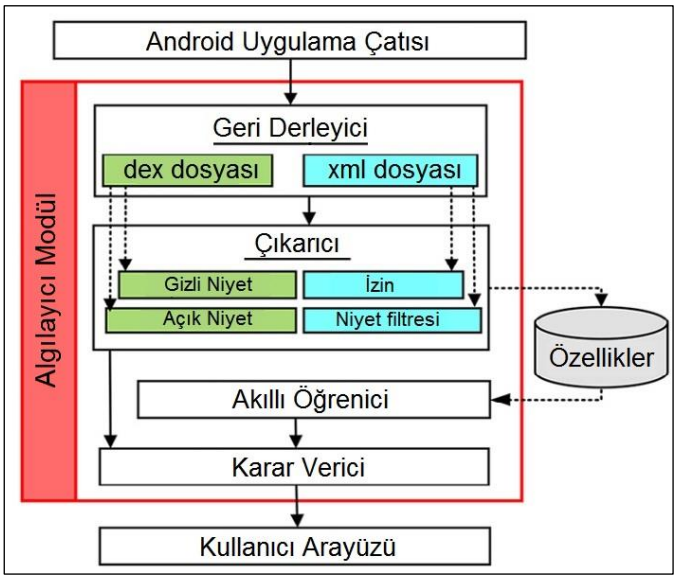
Şekil 3.10. a) Kişiselleştirme kategorisinin izin modeli b) Fotoğrafçılık kategorisinin izin modeli [66]

Önerilen metodolojinin değerlendirilmesi maksadıyla 2013 yılında Google Play uygulama mağazasından her biri 190 ile 590 arasında olmak üzere 35 farklı kategoriden toplam 9512 iyicil uygulama toplanmıştır. Toplanan uygulamalarda talep edilen izinlerden sadece Android işletim sisteminin 4.4 sürümünde bulunan izinlerin kaydı tutulmuştur. Uygulamalar rastgele olarak on kümeye ayrılmış ve dokuz küme eğitim, bir küme test maksadıyla kullanılmıştır. Sınıflandırmada hata olmaması amacıyla kategorilerin uygulama sayıları 300 olarak eşitlenmiştir. Ardından Rastgele Ağaç, Destek Vektör Makineleri ve Naive Bayes sınıflandırma yöntemleri test edilmiş ve Naive Bayes yönteminin en iyi sınıflandırma sonuçlarını sağladığı tespit edilmiştir. Yapılan testlerin sonucunda kategorilerin izin modelleriyle tanımlanabileceği ve bu modellere dayanan bir risk uyarısının kötücül uygulama tespitine olanak sağladığı bulunmuştur.

AndroDialysis: Analysis of Android Intent Effectiveness in Malware Detection

Feizollah ve diğerleri yaptıkları çalışmada Android uygulamalarının AndroidManifest.xml dosyası içerisinde tanımladıkları ya da kod içerisinde kullandıkları açık veya gizli niyetlerin

(intent) kötücül yazılım tespitindeki etkisini incelemek maksadıyla AndroDialysis sistemini geliştirmişlerdir [67]. Geliştirilen sistemin mimarisi Şekil 3.11'de gösterilmektedir.



Şekil 3.11. AndroDialysis sistem mimarisi [67]

Niyetler, uygulama içerisindeki etkinlikler arasında ya da uygulamalar arasında bilgi paylaşımını sağlayan nesnelerdir. Uygulamalar alabilecekleri niyet türlerini niyet filtreleri ile AndroidManifest .xml dosyası içerisinde belirtebilirler. Çalışma esnasında ise açık veya gizli niyet nesneleri ile başka uygulamaya bilgi gönderebilirler ve bir görevi icra etmesini isteyebilirler.

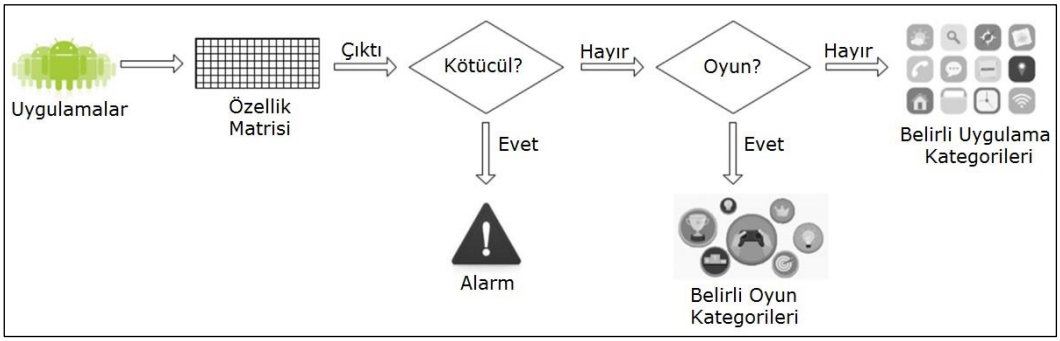
AndroDialysis sisteminde geri derleyici olarak apktool aracı kullanılmıştır. Bu araç ile uygulamaların AndroidManifest.xml dosyaları okunabilir hale getirilmekte ve uygulama kodları smali formatına dönüştürülmektedir. Ardından manifest dosyası içerisinden izinler ve niyet filtreleri, kod içerisinden de açık veya gizli niyet nesneleri çıkarılmaktadır. Bu özellikler akıllı öğreniciye gönderilmekte, burada Bayes Ağları yöntemi kullanılarak veri modeli öğrenilmektedir. Çalışma kapsamında çeşitli tahmin etme ve arama algoritmaları ile testler yapılmıştır. Karar verici bileşeni incelenen uygulamanın özellikleri ile akıllı öğreniciden alınan modeli kullanarak uygulamanın iyicil ya da kötücül olduğunu tespit etmektedir.

Çalışma kapsamında iyicil veri kümesi olarak kullanmak üzere Google Play'deki 27 uygulama ve 17 oyun kategorisinden toplam 1846 uygulama indirilmiş ve Virustotal'de

taratılmıştır. Kötücül veri kümesi olarak ise Drebin kullanılmıştır. Yapılan testlerin sonucunda kötücül yazılımların iyicil yazılımlara göre 7,5 kat daha fazla niyet filtresi tanımladıkları bulunmuştur, bu da kötücül yazılımların daha fazla etkinlik çalma çabasında olduklarını göstermektedir. Ayrıca sadece izinler kullanıldığında %83 doğru pozitif oranına ulaşılabilirken sadece niyetler kullanıldığında %91 doğru pozitif oranına ulaşılabildiği, ikisinin birlikte kullanımıyla %95,5 oranına kadar çıkılabildiği tespit edilmiştir. Sonuçta niyetlerin kötücül yazılım tespitinde izinlere göre daha etkili olduğu ve ikisinin birlikte kullanımıyla başarı oranının oldukça yükseldiği belirtilmiştir.

Detecting Android Malicious Apps and Categorizing Benign Apps with Ensemble of Classifiers

Wang ve diğerleri yaptıkları çalışmada kötücül uygulamaları tespit etme ve iyicil uygulamaları belirlenmiş 24 kategoriden birine otomatik olarak kategorize etme amaçlı bir sistem geliştirmişlerdir [68]. Öncelikle her bir uygulamadan 11 türdeki toplam 2 374 340 adet özellik çıkarılmıştır. Daha sonra sistemin tespit ve kategorize etme etkinliğini artırmak maksadıyla özelliklerin ağırlıklarını sıralamak için Destek Vektör Makineleri kullanılarak en üst sıradaki 34 630 özellik seçilmiştir. Ardından Destek Vektör Makineleri, Rastgele Orman, K-En Yakın Komşu, Sınıflandırma ve Regresyon Ağacı ve Naive Bayes yöntemlerinden oluşan beş sınıflandırıcının birleşimi kullanılarak kötücül uygulamalar tespit edilmiş ve iyicil uygulamalar otomatik olarak kategorize edilmiştir. Yapılan çalışmaya göre oyun uygulamaları ile diğer uygulamalar arasında büyük bir fark bulunmakta olduğundan sistem öncelikle bir uygulamanın oyun olup olmadığını tespit etmekte ve ardından sonuca uygun olarak uygulamaya 8 oyun ya da 16 uygulama kategorisinden birini atamaktadır. Geliştirilen sistemin akış şeması Şekil 3.12'de gösterilmiştir.



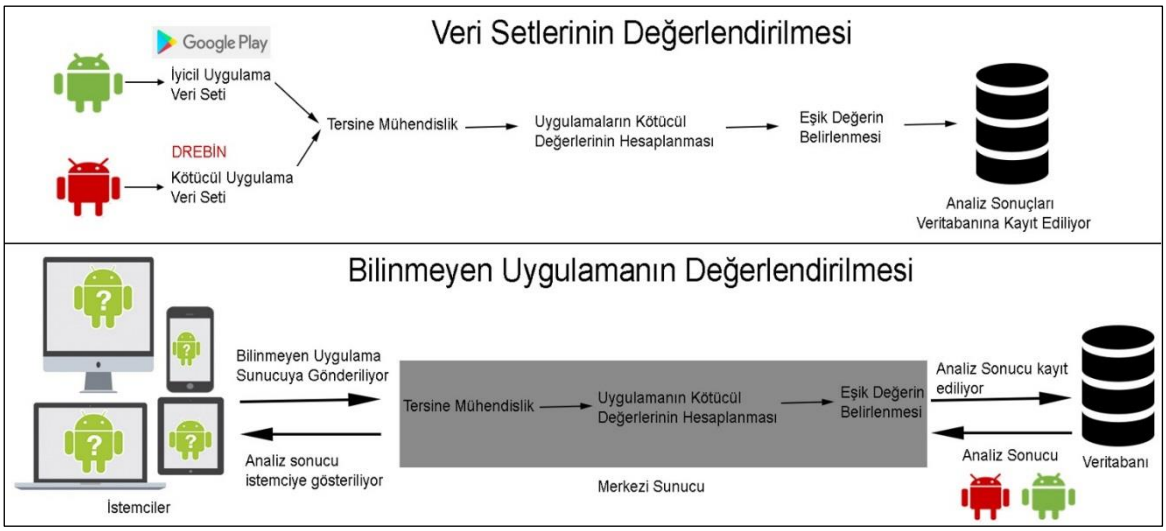
Şekil 3.12. Wang ve diğerlerinin geliştirdiği sistemin akış şeması [68]

Sistemi test etmek maksadıyla Çin'in en büyük uygulama mağazalarından biri olan Anzhi sitesinden 18 866 adet oyun uygulaması, 88 461 adet oyun harici uygulama ve 8701 adet kötücül uygulama kullanılmıştır. İyicil uygulamaların gerçekten iyicil olduklarından emin olmak için uygulamalar VirusTotal ile taratılmış ve hiçbir anti virüs yazılımı tarafından tespit edilmeyen uygulamalar iyicil olarak değerlendirilmiştir. Tüm uygulamaların %90'ı eğitim, %10'u test maksadıyla kullanılmıştır. Yapılan testlerin sonucunda sistem kötücül uygulamaları %99,39'luk bir doğruluk oranıyla tespit edebilmiş ve oyun uygulamalarını %66,23'lük, diğer uygulamaları %82,93'lük bir doğruluk oranıyla kategorize edebilmiştir.

Web Tabanlı Android Kötücül Yazılım Tespit Sistemi

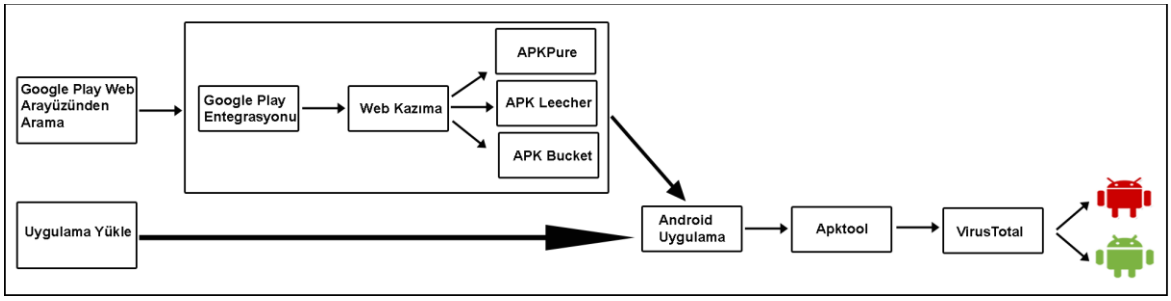
Kiraz yaptığı tez çalışmasında Android işletim sistemine yönelik olarak geliştirilen kötücül yazılımları tespit etmek maksadıyla statik analiz yöntemini kullanan web tabanlı bir sistem geliştirmiştir [69]. Bahse konu sistem önce iyicil ve kötücül uygulamalar ile eğitilmiş, ardından test edilmiştir. İyicil uygulama veri kümesi oluşturmak maksadıyla Google Play mağazasında bulunan en popüler uygulamalar indirilmiş ve VirusTotal taramasından geçirilerek gerçekten iyicil oldukları tespit edilen 1173 uygulama kullanılmıştır. Kötücül uygulama veri kümesi olarak ise Drebin projesinde toplanan 5545 kötücül uygulamadan yararlanılmıştır. Çalışma kapsamında Android API sürümü olarak API 16 seçilmiştir

Eğitim aşamasında iyicil ve kötücül uygulamaların AndroidManifest.xml dosyaları içerisinde talep ettikleri izinler çıkarılmış ve veri tabanına kaydedilmiştir. Test aşamasında ise her bir iznin kötücül uygulamalarda bulunma yüzdesi bulunmuş, ardından iyicil uygulamalarda bulunma yüzdesi kötücül uygulamalarda bulunma yüzdesinden çıkarılarak iznin kötücül değeri hesaplanmıştır. Sonra uygulamanın talep ettiği tüm izinlerin kötücül değerleri toplanarak uygulamanın toplam kötücül değeri elde edilmiştir. Ayrıca VirusTotal'in API'si kullanılarak her bir uygulamanın VirusTotal veri tabanında bulunan sonuçları elde edilmiştir. Tez çalışmasında geliştirilen sistem tarafından hesaplanan uygulama toplam kötücül değeri ile VirusTotal'den elde edilen uygulama kötücül değerinin ortalaması alınarak ilgili uygulamanın nihai kötücül değeri hesaplanmıştır. Geliştirilen sistemin mimarisi Şekil 3.13'te sunulmuştur.



Şekil 3.13. Geliştirilen sistemin mimarisi [69]

Tez çalışması kapsamında geliştirilen sistem istemci sunucu mimarisine göre çalışmakta olup uygulama yüklemeyi ve APKPure, APK Leecher ve APK Bucket sitelerinden web kazıma ile uygulama indirmeyi desteklemektedir. Geliştirilen web uygulamasının çalışma adımları Şekil 3.14'te sunulmuştur.

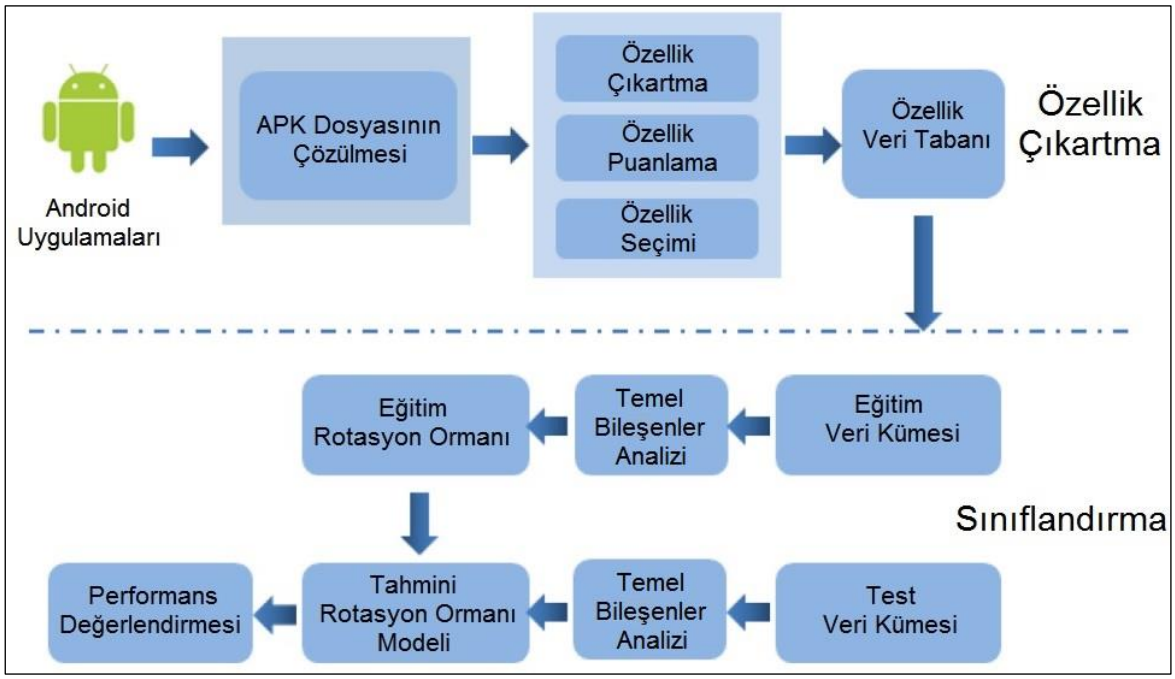


Şekil 3.14. Geliştirilen web uygulamasının çalışma adımları [69]

Tez çalışması kapsamında dört farklı yöntemle test yapılmıştır. Birinci yöntemde sadece varsayılan olarak tanımlı izinlere göre değerlendirme yapılmıştır. İkinci yöntemde varsayılan olarak tanımlı izinlerin yanı sıra uygulamaların tanımladığı yeni izinler de kullanılmıştır. Üçüncü yöntemde sadece varsayılan olarak tanımlı izinlere ve VirusTotal'den elde edilen uygulama kötücül değerine göre, dördüncü yöntemde ise varsayılan olarak tanımlı izinlerin yanı sıra uygulamaların tanımladığı yeni izinlere ve VirusTotal'den elde edilen uygulama kötücül değerine göre değerlendirme yapılmıştır. Test sonuçlarında en başarılı yöntem %97,73'lük doğru tespit başarı oranıyla dördüncü yöntem olmuştur.

DroidDet: Effective and robust detection of android malware using static analysis along with rotation forest model

Zhu ve diğerleri yaptıkları çalışmada Android kötücül uygulamalarını tespit etmek için rotasyon ormanı yöntemine dayanan bir yaklaşım ortaya koymuşlardır [70]. Geliştirdikleri sistem öncelikle uygulamaların talep ettikleri izinleri, hassas API çağrılarını, izlenen sistem olaylarını ve izin sayısı-smali boyutu oranlarını çıkarmaktadır. Ardından çıkarılan özellikler puanlanmakta ve özellik seçimi yapılmaktadır. Temel bileşenler analizi ile birlikte rotasyon ormanı yöntemi kullanılarak bir model oluşturulmakta ve bu model kötücül uygulamaların tespit edilmesi maksadıyla kullanılmaktadır. Geliştirilen sistemin mimarisi ve çalışma mantığı Şekil 3.15'te gösterilmektedir.



Şekil 3.15. Geliştirilen sistemin mimarisi ve çalışma mantığı [70]

Çalışma kapsamında Android resmi uygulama mağazasından 1065 adet iyicil ve VirusShare [71] sitesinden 1065 adet kötücül uygulama indirilmiştir. Bunlardan 600 adet iyicil ve 600 adet kötücül uygulama eğitim amacıyla, geriye kalan uygulamalar ise test amacıyla kullanılmıştır. Yapılan testlerin sonucunda ortalamada %88,26 doğruluk, %88,40 hassasiyet ve %88,16 duyarlılık oranları elde edilmiştir. Ayrıca aynı şartlarda destek vektör makineleri yöntemi kullanılarak testler de yapılmış ve rotasyon ormanı yönteminin başarı oranlarının daha yüksek olduğu ortaya çıkarılmıştır.

An Android mutation malware detection based on deep learning using visualization of importance from codes

Yen ve Sun yaptıkları çalışmada Android kötücül yazılımlarını tespit etmek üzere uygulama kodları içerisinde geçen kelimelerin önem değerlerine dayanan bir yaklaşım ortaya koymuşlardır [72]. Geliştirdikleri sistem öncelikle dex2jar ve jad araçlarını kullanarak apk dosyalarını java kaynak kodlarına dönüştürmektedir. Ardından terim frekansı – ters belge frekansı isimli istatistik yöntemi kullanılarak kodlar içerisinde geçen kelimelerin önem değerleri belirlenmiştir. Daha sonra SimHash ve djb2 algoritmaları kullanılarak uygulamalara ilişkin resimler oluşturulmaktadır. Şekil 3.16'da bu şekilde oluşturulmuş bir resim gösterilmektedir.

Şekil 3.16. Oluşturulan bir uygulama resmi örneği [72]

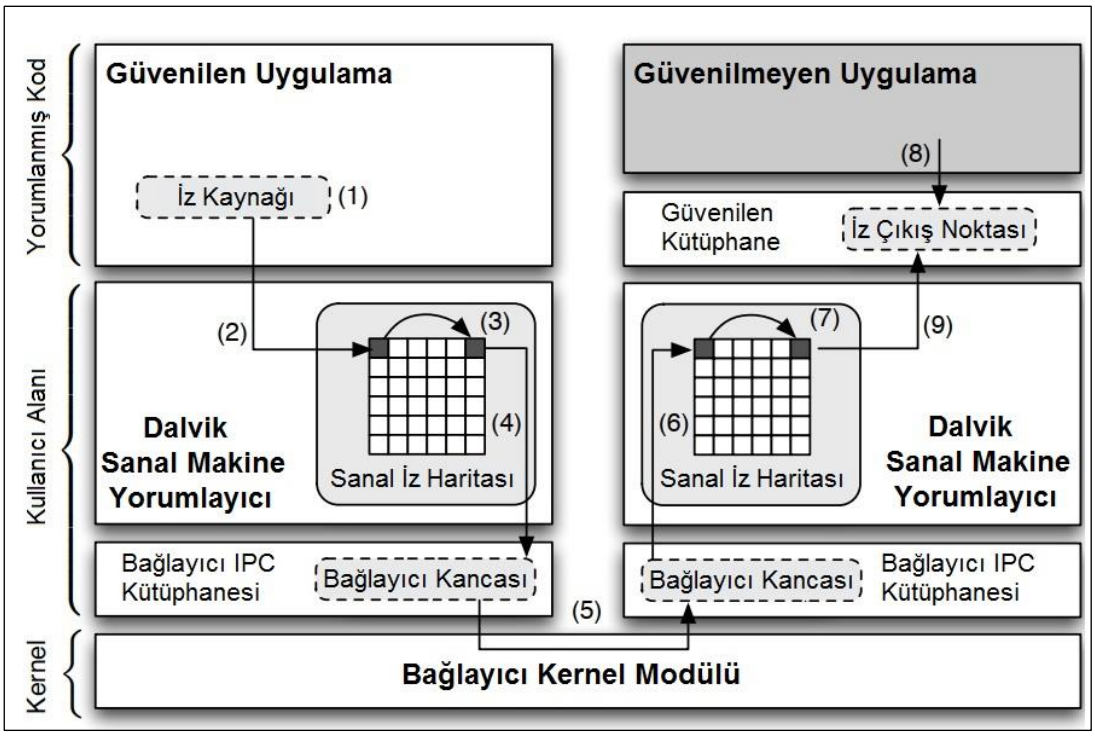
Uygulama resimleri oluşturulduktan sonra Berkeley AI Research tarafından geliştirilen bir evrişimli sinir ağları çatısı olan Caffe derin öğrenme çatısı [73] kullanılarak sistem eğitilmiş ve test edilmiştir. Çalışma kapsamında 720 iyicil ve 720 kötücül uygulama toplanmış, bunlardan 500'er adedi eğitim, 220'şer adedi ise test için kullanılmıştır. Yapılan testlerin sonucunda %92 doğruluk oranına ulaşılmıştır.

3.2. Dinamik Analiz

Dinamik analiz yönteminde APK uygulaması cihaz ya da emülatör üzerinde çalıştırılır ve çalışması sırasında yaptığı davranışlar izlenir. Bu bölümde dinamik analizi yöntemini kullanan çalışmalar incelenecektir.

TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones

Enck ve diğerleri yaptıkları çalışmada hassas veri kaynaklarından çıkan verileri etiketleyen, bunları sistem genelinde dinamik olarak takip ve analiz edebilen TaintDroid sistemini geliştirmişlerdir [74]. Sistemin mimarisi ve çalışma mantığı Şekil 3.17'de gösterilmektedir.



Şekil 3.17. TaintDroid sisteminin mimarisi ve çalışma mantığı [74]

Sistem kişisel hassas kaynaklardan çıkan verilere otomatik olaak etiket atamakta, veri uygulama değişkenleri, dosyalar ve işlemler arası mesajlarda ilerledikçe yeni etiketler uygulanarak takip edilmektedir. Etiket atanmış herhangi bir verinin cihazdan çıkmaya çalıştığında TaintDroid verinin etiketlerini, veriyi çıkartan uygulamayı ve verinin hedefini kayıt altına almaktadır.

Uygulamanın değerlendirilmesi maksadıyla konum, kamera ve mikrofon verilerini kullanan rastgele seçilmiş 30 adet uygulama kullanılmıştır. Yapılan testlerde bu uygulamalardan 20 adedinin hassas verileri şüpheli bir şekilde işledikleri, 15 adedinin kullanıcı konumunu uzak reklam sunucularına aktardığı ve 7 adedinin cihaz bilgilerini topladığı bulunmuştur. Ayrıca sistemin yaklaşık %14 oranında bir işlemci yüküne ve yaklaşık %4,4 oranında bellek artışına yol açtığı tespit edilmiştir.

MockDroid: Trading Privacy for Application Functionality on Smartphones

Beresford ve diğerleri yaptıkları çalışmada kullanıcılara uygulamaların belirli kaynaklara erişimlerini kısıtlama imkânı sunan MockDroid sistemini geliştirmişlerdir [75]. Bu sistem sayesinde herhangi bir uygulama bir kaynağa erişmeye çalıştığında kullanıcılar uygulamaya boş veya sahte veriler verilmesini ya da kaynağın erişilemez olduğunun bildirilmesini sağlayabilmektedir. Uygulamalara boş veya sahte veriler verilebilecek kaynaklar arasında konum bilgisi, internet, mesajlar, takvim, rehber, cihaz bilgileri vb. bulunmaktadır. Böylece uygulamaların işlevselliğinde meydana gelebilecek küçük bir kayba karşılık istenmeyen veri sızıntıları engellenebilmekte ve cihaz telefon üzerindeki işlemleri kontrol altına alınabilmektedir.

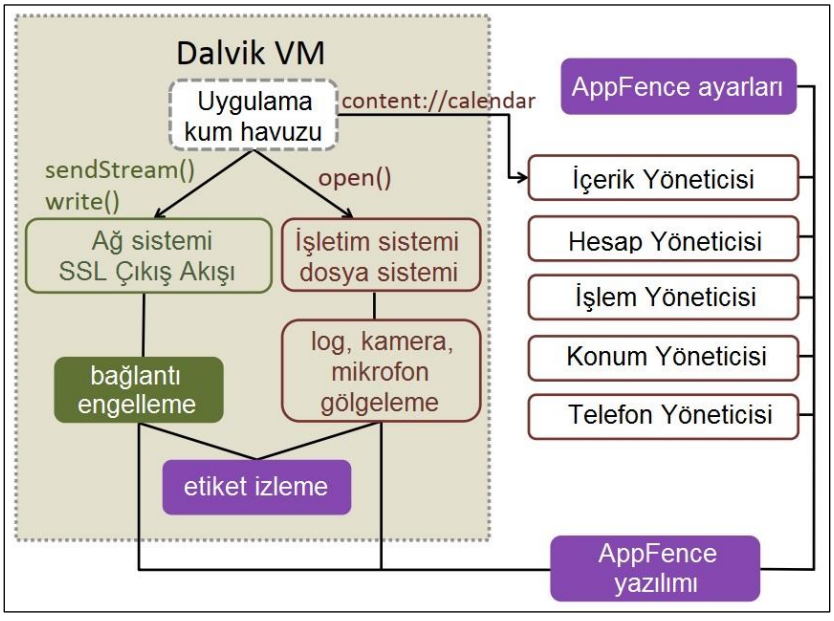
MockDroid sisteminin prototipi Android 2.2.1 işletim sistemi sürümü üzerinde kurulmuştur. Sistemin uygulamalara çalışma esnasında müdahale edebilmesini sağlamak için her bir tehlikeli API çağrısı öncesinde çalıştırılan izin kontrol mekanizmasında değişiklik yapılmıştır. Değiştirilen izin kontrol mekanizması herhangi bir uygulama çalışma esnasında bir izin istediğinde önce kurulum sırasında uygulamanın bu izni talep edip etmediğini kontrol etmektedir. Ardından izin talep edilmiş olması halinde kullanıcının bu izinle ilgili herhangi bir kural belirleyip belirlemediği kontrol edilmektedir. Böylece kaynaklara erişimle ilgili olarak uygulamalar arasında farklı kurallar belirlenebilmektedir.

Geliştirilen uygulamanın değerlendirilmesi maksadıyla TaintDroid çalışmasında kullanılan ve konum bilgisine erişim sağlayan 23 adet uygulama kullanılmıştır. MockDroid sistemi aktif ve pasifken bu uygulamaların çalışmaları incelenmiştir. Yapılan incelemelerde tüm uygulamaların konum ve internet erişimlerinde sahte veriler verilebilmiş ve uygulamaların işlevselliğinde önemli bir kayıp gözlenmemiştir.

These Aren't the Droids You're Looking For: Retrofitting. Android to Protect Data from Imperious Applications

Hornyack ve diğerleri tarafından geliştirilen AppFence sistemi, TaintDroid sistemini geliştirerek kullanıcıların özel olarak değerlendirdikleri bilgilerden elde edilen verileri izlemekte ve bu verilerden yapılacak istenmeyen iletimleri engellemektedir [76]. Bunun için hassas verileri gölgeleme ya da bu verilerin cihazdan çıkışını engelleme şeklinde iki farklı yaklaşım sunmaktadır. İlk yaklaşımda MockDroid'e benzer şekilde özel verilerin yerine sahte veriler sağlanarak kötücül yazılımların özel verileri çalmalarını engellenmektedir. İkinci yaklaşımda bu veriler ağ üzerinden dışarıya çıkartılmaya çalışıldığında bağlantı engellenerek veriler korunmaktadır.

Çalışma kapsamında öncelikle 2010 yılının Kasım ayında Android Market'teki 22 kategorinin her birinden 50'şer uygulama olmak üzere toplamda 1100 uygulama indirilmiştir. Bu uygulamaların incelenmesi neticesinde 12 adet hassas veri kaynağına erişmek için zorunlu olan 11 adet izin belirlenmiştir. Ardından her bir izin için onu kullanan 10'ar uygulama olmak üzere toplam 110 uygulamadan oluşan bir alt küme çıkarılmıştır. Bu uygulamalar elle yaklaşık beş dakika boyunca çalıştırılarak davranışları incelenmiştir. Daha sonra Android işletim sistemindeki yaklaşık 5000 satır kodda değişiklik yapılarak geliştirilen AppFence sistemi Android işletim istemine eklenmiştir. AppFence sistem mimarisi Şekil 3.18'de gösterilmektedir.

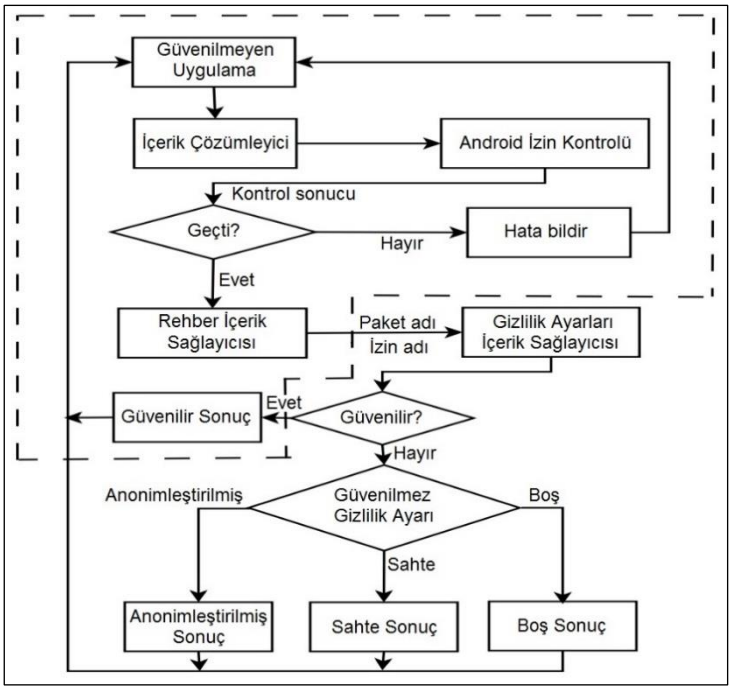


Şekil 3.18. AppFence sisteminin mimarisi [76]

Geliştirilen sistemi değerlendirmek için 110 uygulamadan oluşan alt kümede bulunan 50 uygulama seçilmiş ve bunlar için otomatik test betikleri yazılmıştır. Yapılan testlerde uygulamaların %66'sında herhangi bir yan etki olmadan gizlilik kontrolü sağlanabildiği ancak diğer %34'ünün ise AppFence'in zorladığı gizlilik kontrollerinin ihlalini gerektiren işlevlere sahip oldukları tespit edilmiştir.

Taming Information-Stealing Smartphone Applications (on Android)

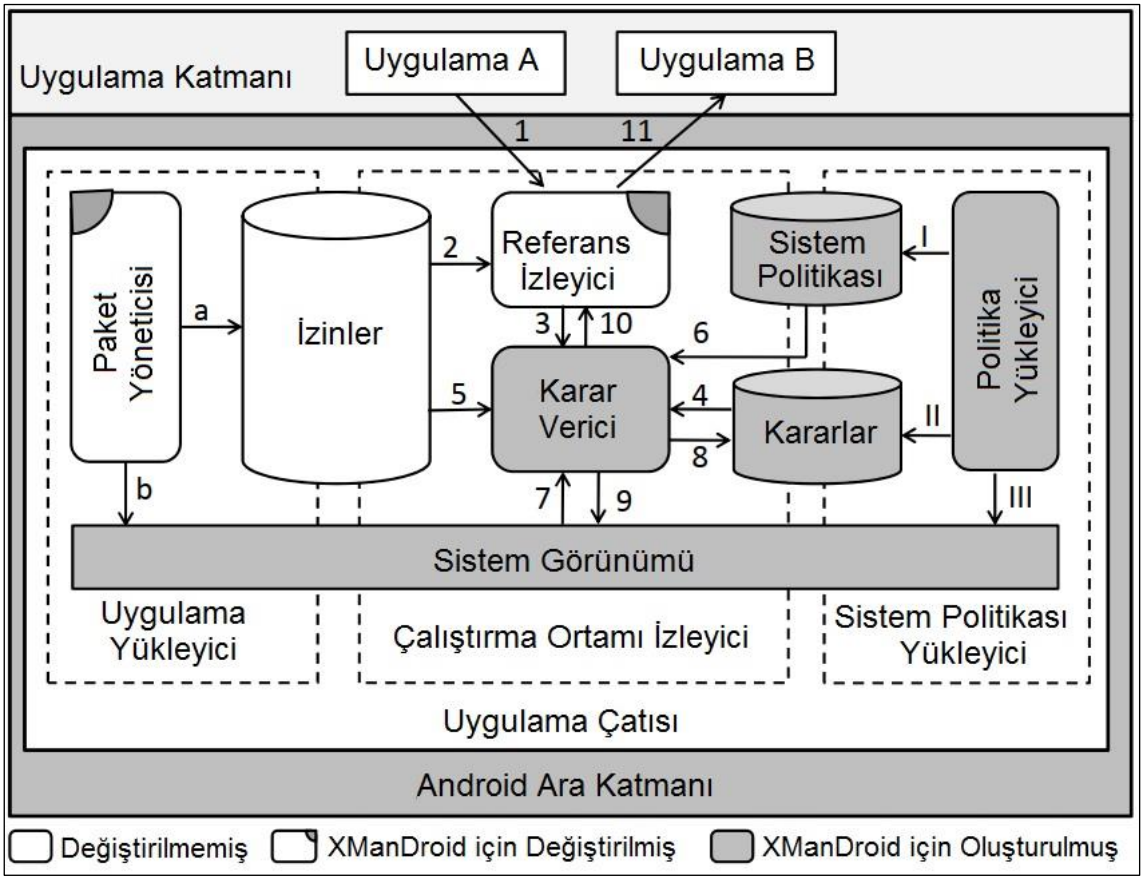
Zhou ve diğerleri yaptıkları çalışmada Android işletim sistemi üzerinde uygulamaların çalışması sırasında belirli kaynaklara erişimlerini dinamik olarak yönetebilmek amacıyla TISSA sistemini geliştirmişler ve Android işletim sistemi içerisine eklemişlerdir [77]. TISSA sistemi MockDroid ve AppFence sistemlerine benzer şekilde uygulamaların hassas veri kaynaklarına erişimlerine çalışma zamanında müdahale etmektedir. Bu amaçla kullanıcılara dört farklı seçenek sunulmaktadır. Boş seçeneğinde erişim talep eden uygulamaya hiçbir veri verilmez, talep edilen veriler yokmuş gibi davranılır. Anonim seçeneğinde kullanıcı verilerinin anonimleştirilmiş hali verilir. Sahte seçeneğinde talep edilen bilgilerin yerine sahte veriler verilir. Güvenilir seçeneğinde talep edilen bilgiler uygulamaya aynen verilir. Örnek olarak bir uygulamanın rehbere erişiminde TISSA sisteminin çalışma mantığı Şekil 3.19'da gösterilmektedir.



Şekil 3.19. TISSA sisteminin çalışma mantığı [77]

XManDroid: A New Android Evolution to Mitigate Privilege Escalation Attacks

Bugiel ve diğerleri yaptıkları çalışmada mobil uygulamalar arasındaki iletişimi çalışma zamanında izleyip analiz eden ve uygulamaların kullanıcılar tarafından oluşturulmuş erişim kurallara uygun davranmalarını sağlayan XManDroid sistemini geliştirmişlerdir [78]. XManDroid, bileşenler arası iletişim (ICC) yoluyla uygulamalar arasında yapılan veri iletişimini takip eder, inceler ve belirlenmiş politikalara göre iletişime izin verilip verilmeyeceğine karar verir. Örneğin "Rehber bilgilerine erişim hakkı olan bir uygulama ağ erişim hakkına sahip bir uygulama ile iletişim kuramaz" şeklinde bir politika belirlenebilir ve XManDroid tarafından bu politika zorlanabilir. Bu şekilde belirli izinler verilmiş iyicil uygulamalardaki zafiyetleri kullanan kötücül uygulamaların bu iyicil uygulamaları sömürmesinin önüne geçilmeye çalışılmaktadır. Aynı zamanda ayrı ayrı olarak tehlikeli oluşturmayacak izinlere sahip uygulamaların birlikte çalışarak cihaz üzerinde tehlikeli işlemler yapmaları da engellenmektedir. XManDroid sisteminin mimarisi ve çalışma mantığı Şekil 3.20'de gösterilmektedir.

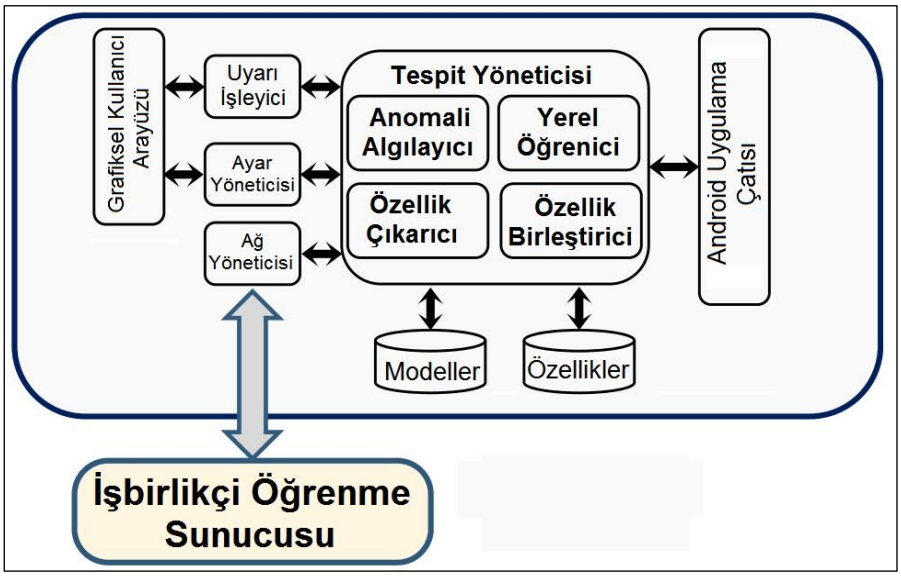


Şekil 3.20. XManDroid mimarisi ve çalışma mantığı [78]

Geliştirilen XManDroid sisteminde çalışma sırasında referans izleyici bir ICC çağrısı yapıldığında araya girip uygulamanın izin atamalarını kontrol eder. ICC çağrısına izin vermezse işlem bitirilir. İzin verilmesi halinde karar verici devreye girer. Eğer daha önce bu ICC çağrısı yapılmış ve bu çağrıya ilişkin bir karar mevcutsa aynı karar tekrarlanır. Eğer bir karar mevcut değilse izinler, sistem politikası, sistem görünümü bileşenlerinden gerekli bilgiler alınır. İşlem sonucunda verilen ICC çağrısına ilişkin karar, kararlar bileşenine eklenir. Eğer karar olumluysa sistem görünümü güncellenir. Ayrıca verilen karar referans izleyiciye bildirilir ve referans izleyici karara göre ICC çağrısına izin verir ya da çağrıyı engeller.

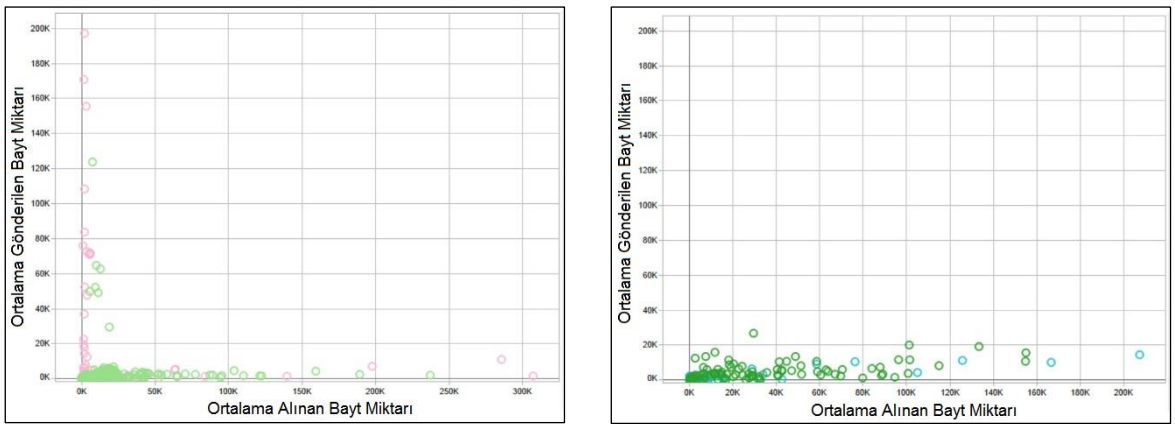
Mobile malware detection through analysis of deviations in application network behavior

Shabtai ve diğerleri yaptıkları çalışmada Android işletim sistemi yüklü cihaz üzerinde çalışan ve uygulamaların çalışmaları esnasında normal ağ davranış modellerini cihaz üzerinde çıkartan bir sistem geliştirmişlerdir [79]. Sistem ortalama gönderilen/alınan bayt miktarları ve bunların yüzdeleri, gönderme/alma zaman aralıkları vb. uygulama seviyesi ağ özelliklerini toplamakta, REPTree ve Karar Tablosu gibi çeşitli makine öğrenmesi algoritmalarını kullanarak uygulamaların ağ modellerini belirlemekte ve öğrenmektedir. Ardından öğrenilen ağ modelinden yapılan sapmalar incelenerek kötücül yazılımlar tespit edilebilmektedir. Geliştirilen sistemin mimarisi ve çalışma mantığı Şekil 3.21'de gösterilmektedir.



Şekil 3.21. Geliştirilen sistemin mimarisi ve çalışma mantığı [79]

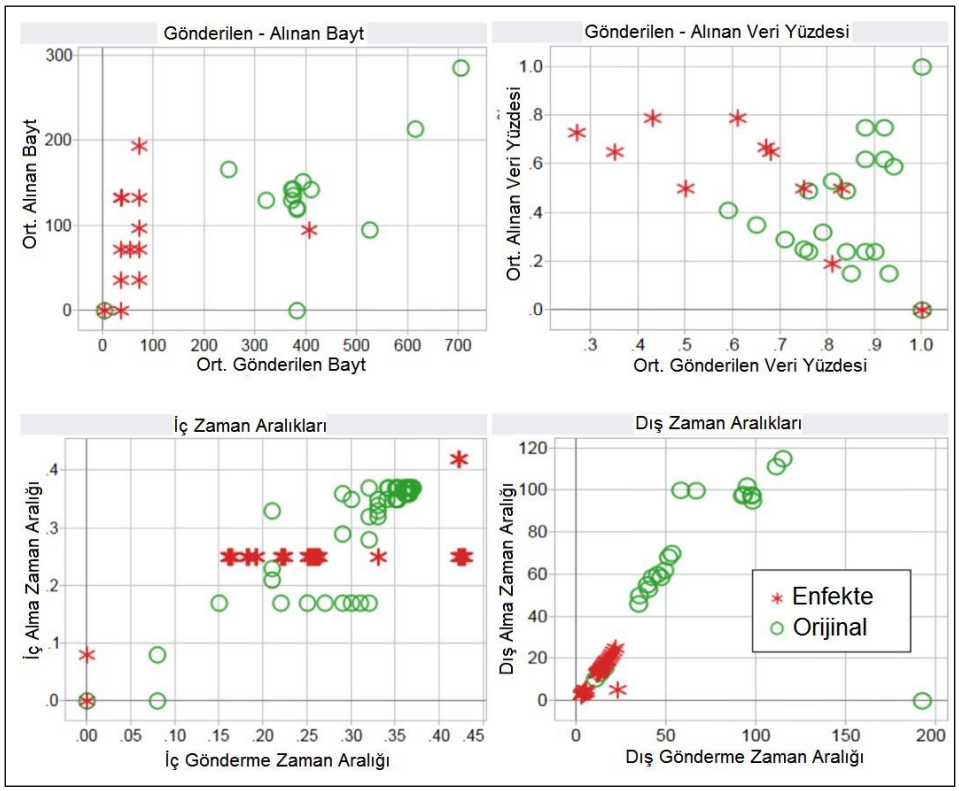
Yapılan çalışmada farklı türdeki uygulamaların farklı ağ modellerine sahip olacağı buna karşın aynı türe ait farklı uygulamaların benzer ağ modeline sahip olacağı iddia edilmiştir. Bu kapsamda öncelikle Facebook, Skype, Gmail ve WhatsApp uygulamalarının ağ modelleri belirlenmiş ve oldukça farklı oldukları gösterilmiştir. Ardından e-posta istemcisi türüne ait Gmail uygulaması ile Android'in yerel e-posta istemcisinin ağ modelleri (Şekil 3.22a) ve internet tarayıcısı türüne ait Mozilla Firefox uygulaması ile cihazın yerel internet tarayıcısının ağ modelleri (Şekil 3.22b) belirlenmiş ve benzer oldukları ortaya çıkartılmıştır.



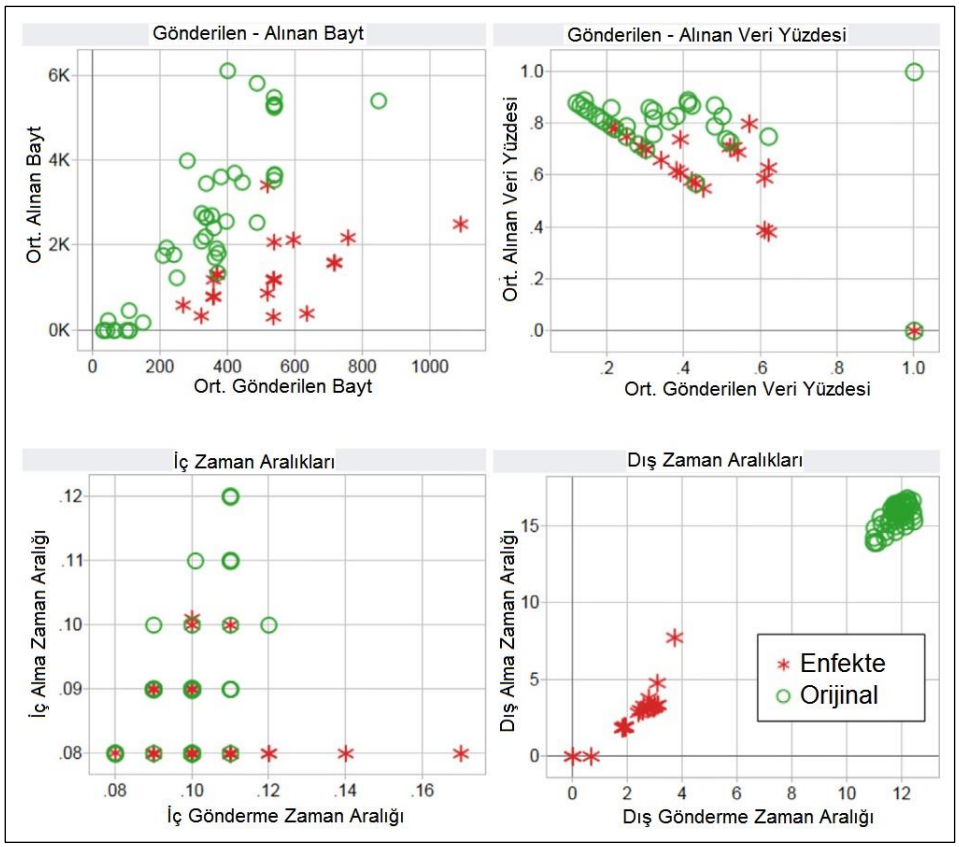
a b

Şekil 3.22. Aynı türdeki farklı uygulamaların ağ modeli karşılaştırmaları a) E-posta istemci uygulamaları b) İnternet tarayıcı uygulamaları [79]

Ardından geliştirilen sistem ile kötücül uygulamaların tespit edilebileceğini sergilemek amacıyla ShotGun uygulamasının iyicil hali ile Geinimi Trojan enjekte edilmiş kötücül halinin ağ modelleri (Şekil 3.23a) ve K-9 e-posta istemci uygulamasının iyicil hali ve kendi yazdıkları kötücül kod enjekte edilmiş kötücül halinin ağ modelleri (Şekil 3.23b) belirlenmiş ve karşılaştırılmıştır. Karşılaştırmalarda ortalama alınan ve gönderilen bayt miktarları, ortalama alınan ve gönderilen veri yüzdeleri, belirli süre içerisindeki alma ve gönderme zaman aralıkları, belirli süre dışındaki alma ve gönderme zaman aralıkları dikkate alınmıştır. Yapılan karşılaştırmaların sonucunda iyicil uygulamalar ile kötücül uygulamaların ağ özelliklerinde ve dolayısıyla ağ modellerinde büyük farklılıklar olduğu ve bu yöntemle kötücül uygulamaların tespit edilebileceği değerlendirilmiştir. Ayrıca yapılan testler sonucunda sistemin cihaz üzerinde çalışmasına rağmen işlemci ve bellek kullanımı bakımından cihaza önemli bir performans yükü getirmediği bulunmuştur.



a

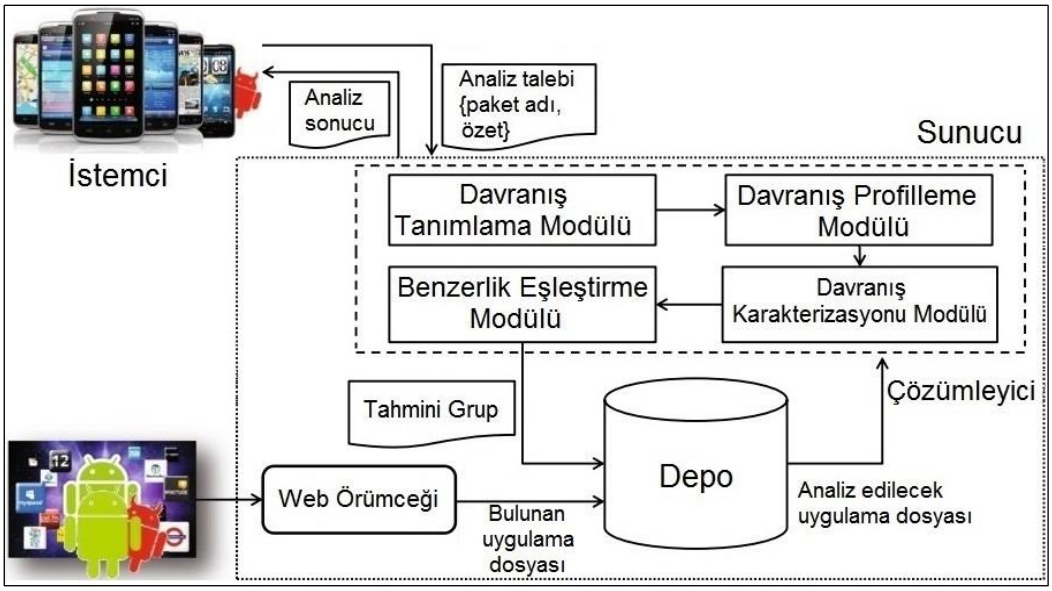


b

Şekil 3.23. Aynı uygulamaların orijinal ve kötücül kod enfekte edilmiş hallerinin ağ modeli karşılaştırmaları a) ShotGun b) K-9 e-posta istemcisi [79]

Detecting and classifying method based on similarity matching of Android malware behavior with profile

Jang ve diğerleri yaptıkları çalışmada Andro-profiler isimli davranış profili belirleme sistemi geliştirmişlerdir [80]. Sistem mobil cihaz üzerinde çalışan bir istemci uygulaması ile analiz ve profil belirleme işlemlerini yapan bir uzak sunucudan oluşmaktadır. İstemci uygulaması cihaz üzerinde bulunan incelenecek uygulamaya özel bilgileri sunucuya göndermektedir. Eğer sunucu bu bilgileri kullanarak uygulamayı internette bulamazsa istemci uygulamanın apk dosyasını sunucuya göndermektedir. Sunucu ise bütünleşik sistem loglarını oluşturmak amacıyla kötücül yazılımları bir emülatör üzerinde çalıştırmakta ve oluşan logları analiz ederek davranış profillerini çıkarmaktadır. Ardından ağırlıklı benzerlik eşleştirme tekniğini kullanarak kötücül yazılım aileleri için oluşturulan temsili davranış profilleri ile karşılaştırmakta ve ait olduğu kötücül yazılım ailesine sınıflandırmaktadır. Geliştirilen Andro-profiler sisteminin mimarisi ve çalışma mantığı Şekil 3.24'te gösterilmektedir.



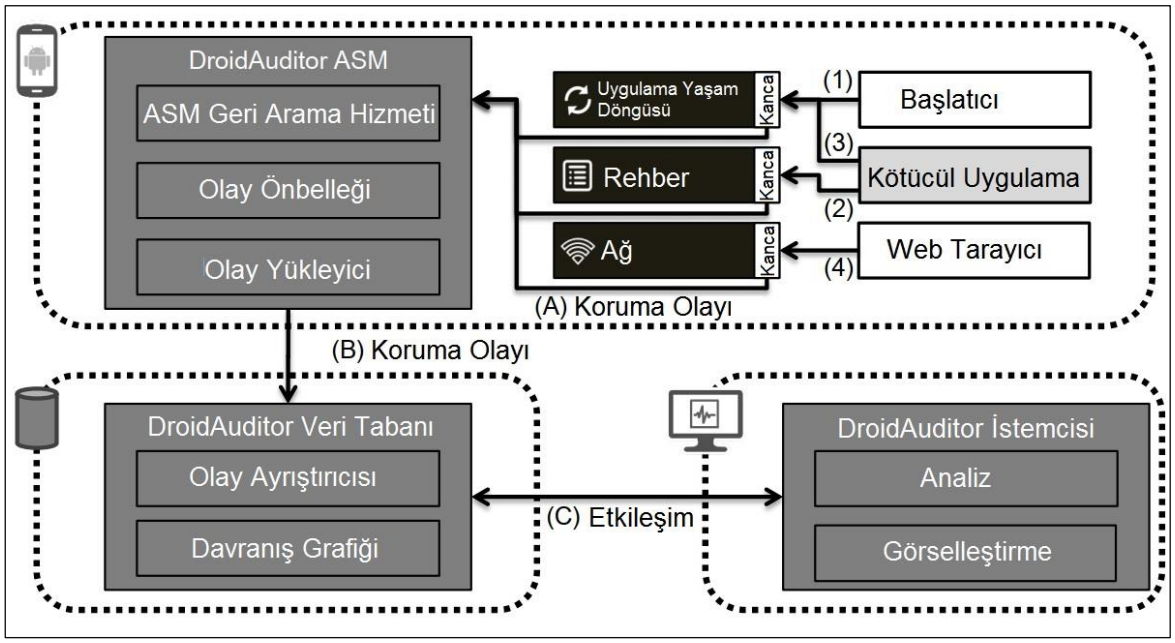
Şekil 3.24. Andro-profiler sistem mimarisi [80]

Sistem kötücül yazılım karakterinin belirlenmesi maksadıyla yüklenebilir kernel modülü (LKM) tarafından sağlanan sistem çağrıları ile bunların argümanlarını ve DroidBox [81] tarafından sağlanan SMS, arama kaydı ve ağ girdi çıktıları gibi sistem kayıtlarını kullanmaktadır. Bu kapsamda kötücül yazılımların eşsiz kötücül davranış modellerine sahip oldukları ve bu davranışların sistem çağrıları ile belirlenebileceği kabul edilmiştir.

Geliştirilen sistemin test edilmesi amacıyla 2013 yılı Ocak ile Ağustos ayları arasında beş farklı kötücül yazılım ailesinden toplam 643 adet kötücül yazılım toplanmıştır. İyicil uygulamalar olarak aynı dönemde Google Play'den indirilen yüksek puanlı 8840 adet uygulama kullanılmıştır. Yapılan testlerin sonucunda Andro-profiler sisteminin %98 sınıflandırma doğruluk oranına ulaştığı ve yeni çıkan kötücül yazılım örneklerini de tespit edebildiği bulunmuştur.

DroidAuditor: Forensic Analysis of Application-Layer Privilege Escalation Attacks on Android

Heuser ve diğerleri yaptıkları çalışmada uygulama seviyesindeki hak yükseltme saldırılarının önüne geçmek maksadıyla DroidAuditor aracını geliştirmişlerdir [82]. Bu araç Android Security Modules (ASM) [83] erişim kontrol mimarisini kullanarak uygulamaların Android işletim sisteminin tüm katmanlarındaki davranışlarını takip etmekte, davranış analizi yapmakta ve etkileşimli davranış grafiklerini çıkarmaktadır. DroidAuditor sistem mimarisi Şekil 3.25'te gösterilmektedir.



Şekil 3.25. DroidAuditor sistem mimarisi [82]

DroidAuditor aracının ASM, veri tabanı ve istemci olmak üzere üç temel bileşeni bulunmaktadır. Cihaz üzerinde ASM çatısı öncelikle güvenlik veya gizlilik bakımından kritik kernel ve orta katman seviyesindeki tüm işletim sistemi bileşenlerine kancalar

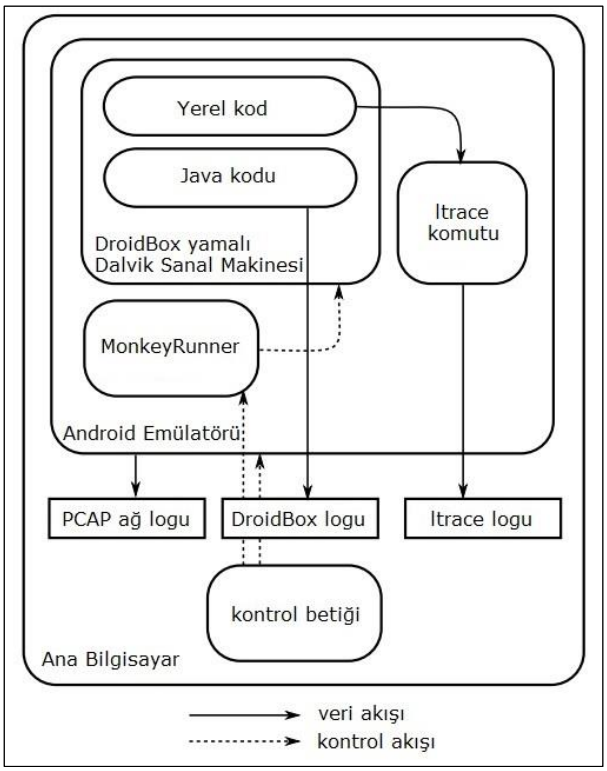
yerleştirmektedir. Cihaz üzerinde çalışan herhangi bir uygulama adım 1-4 arasındaki gibi güvenlik veya gizlilik bakımından kritik bir kaynağa erişim sağlamaya çalıştığı zaman ASM çatısı DroidAuditor ASM bileşenine bildirim göndermektedir. Bu bileşen de bu bildirimi şifreli bir kanal üzerinden veri tabanına iletmektedir. İletilen davranışlar veri tabanında ayrıştırılmakta ve davranış grafikleri halinde saklanmaktadır. Güvenlik analistleri DroidAuditor istemcisini kullanarak etkileşimli davranış grafiklerine ulaşabilmekte ve bunlar üzerinde sorgulama yapabilmektedir.

3.3. Hibrit Analiz

Hibrit analiz yöntemi statik analiz ile dinamik analizin birleşimi olup bu yöntemde APK uygulaması hem statik hem de dinamik analiz teknikleriyle incelenerek uygulama hakkında daha fazla bilgi elde edilmeye çalışılır. Bu bölümde hibrit analiz yöntemini kullanan çalışmalar incelenecektir.

Mobile-Sandbox: Having a Deeper Look into Android Applications

Spreitzenbarth ve diğerleri tarafından geliştirilen Mobile-Sandbox sistemi statik analiz bölümünde VirusTotal hizmetinden faydalanarak anti virüs taraması yaptırmakta, Android SDK içerisinde gelen aapt aracını kullanarak AndroidManifest.xml dosyası içerisinden bilgi toplamakta ve uygulama kodunu çözerek içerisinde incelemeler yapmaktadır [84]. Uygulamanın kötücül olması halinde hangi aileye ait olduğu Kaspersky tarafından yapılan sınıflandırmaya göre belirlenmektedir. Dinamik analiz bölümünde ise uygulama Google tarafından sağlanan Android emülatörü içerisinde çalıştırılarak uygulama tarafından Dalvik sanal makinesinde ve yerel kütüphanelerde gerçekleştirilen tüm işlemler kaydedilmektedir. Tüm kayıtları tutabilmek için TaintDroid/DroidBox sistemi temel olarak kullanılmış ve DroidBox'ın desteklediği sürümü yükseltmek için yama yapılmıştır. Yerel kodları takip edebilmek amacıyla ltrace komutunun değiştirilmiş bir hali kullanılmıştır. Ayrıca uygulamanın tüm ağ iletişim trafiği PCAP logu içerisine kaydedilmiştir. Uygulama ile kullanıcı etkileşimini simüle etmek amacıyla Android SDK ile birlikte gelen MonkeyRunner'dan faydalanılmıştır. Mobile-Sandbox sisteminin dinamik analiz bileşeni Şekil 3.26'da gösterilmektedir.



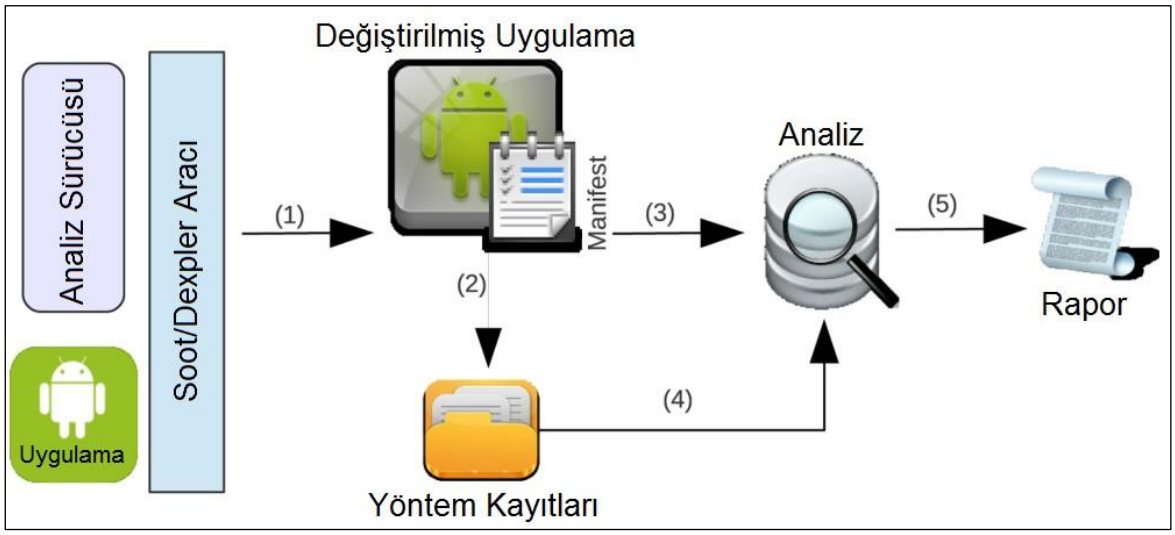
Şekil 3.26. Mobile-Sandbox sisteminin dinamik analiz bileşeni [84]

Geliştirilen sistemin test edilmesi maksadıyla 2011 yılı Aralık ayı ile 2012 yılı Ağustos ayı arasında Asya'daki en önemli uygulama mağazalarından yaklaşık 80 000 uygulama toplanmıştır. Ayrıca farklı ailelerden 7500 kötücül uygulama örneği de toplanmıştır. Ardından Asya'daki uygulama mağazalarından indirilen uygulamalar arasından rastgele seçilen 36 000 uygulama ile kötücül uygulamalar arasından rastgele seçilen 4000 uygulama kullanılarak Mobile-Sandbox sistemi test edilmiştir. Test sonucunda toplamda 4641 uygulamanın kötücül olduğu tespit edilmiş ve bunun sonucunda Asya'daki uygulama mağazalarından indirilen uygulamalar arasından en az 641 adedinin kötücül olduğu bulunmuştur.

Geliştirilen Mobile-Sandbox sisteminde analiz işlemlerinin oldukça uzun sürmesi sistemin olumsuz yanlarından biridir. Sistemde ortalama 3 saniyede anti virüs kontrolü yapılmakta, ardından 8 – 15 saniye arasında statik analiz tamamlanmaktadır. Dinamik analiz işleminde emülatörü temiz bir halde yeniden başlatmak yaklaşık 2 dakika, analiz edilecek uygulamayı kurmak dosya boyutuna bağlı olarak 2 – 6 dakika arası ve uygulama ile MonkeyRunner betiklerinin çalıştırılması istenen kullanıcı olayı sayısına bağlı olarak 6 – 10 dakika arası zaman almaktadır. Ardından 10 saniyede tüm log ve ağ dosyaları analiz edilmektedir.

A Permission Verification Approach for Android Mobile Applications

Geneiatakis ve diğerleri yaptıkları çalışmada gereğinden fazla izinlere sahip iyicil uygulamaların kötücül yazılımlar tarafından sömürülebileceğini, saldırganların bu yolla kötücül faaliyetlerini gizleyebileceklerini ve saldırı tespit sistemlerini atlatabileceklerini değerlendirmişlerdir [85]. Buna bağlı olarak uygulamaların aşırı izin taleplerini tespit etmek maksadıyla Android işletim sisteminin kaynak koduna müdahale etmeyi gerektirmeyen, uygulamaların yeniden paketlenmesine dayanan bir yaklaşım geliştirmişlerdir. Bu yaklaşımda incelenen uygulamanın AndroidManifest.xml dosyasında talep ettiği izinler statik analizle, bu izinlerin çalışma esnasında kullanımı dinamik analizle belirlenmektedir. Bu amaçla Dexpler aracı kullanılmıştır. Dexpler, Bartel ve diğerleri tarafından geliştirilen, Android mobil uygulamalarını analiz etmek, değiştirmek ve yeniden paketlemek için kullanılan Soot çerçevesi tabanlı bir araçtır [86]. Bu çerçevede herhangi bir Android uygulaması uygun şekilde tasarlanmış bir analiz sürücüsüne girdi olarak verilebilmektedir. Bunun sonucunda uygulamanın çözülmüş kodu statik incelemeye tabi tutulabilmekte ve gereksinimlere göre uygulamaya yeni işlevler eklenebilmektedir. İncelenmek istenen uygulama önce Dexpler ile Jimple formatına dönüştürülmekte, böylece Soot çerçevesinin kullanımı mümkün hale gelmektedir. Ardından uygulama kodu içerisine küçük izleme kod parçacıkları yerleştirilmekte ve uygulama yeniden paketlenip imzalanmaktadır. Çalışma tarafından önerilen izin doğrulama yaklaşımının çalışma mantığı Şekil 3.27'de gösterilmektedir.



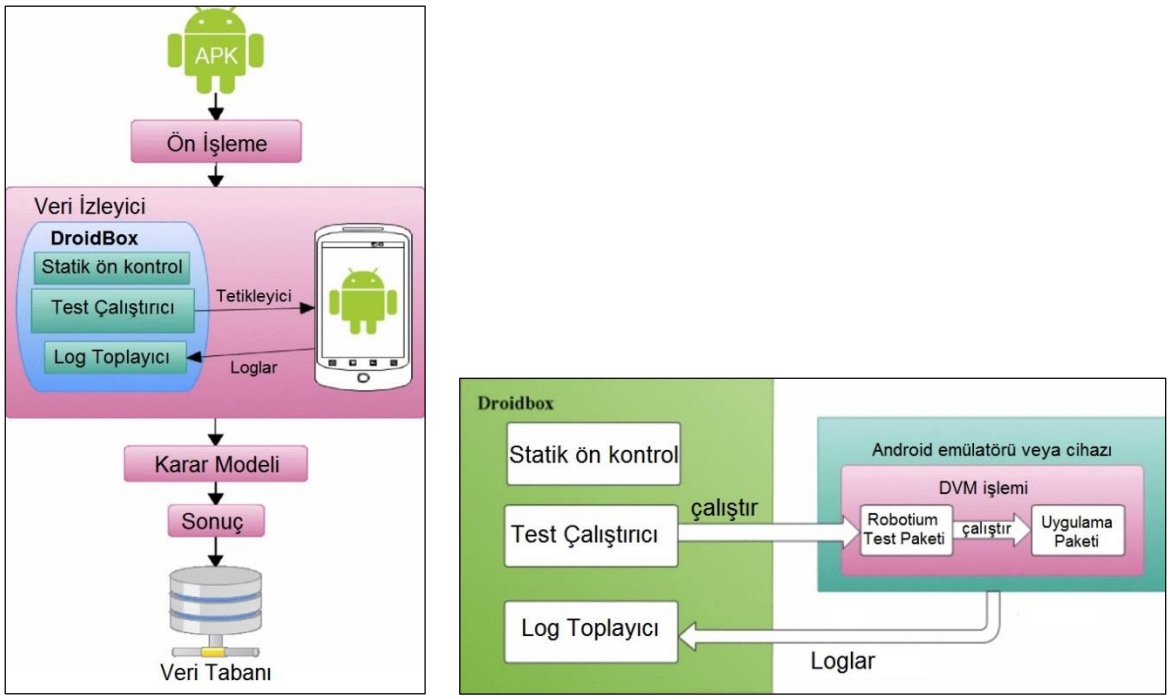
Şekil 3.27. Çalışma tarafından önerilen izin doğrulama yaklaşımı [85]

Öncelikle Soot/Dexpler aracı uygulamayı analiz eder ve çalıştırılan yöntemleri izlemek amacıyla uygulamanın içerisine küçük izleme kod parçacıkları ekler (1). Uygulama her çalıştığında tüm yöntemler kaydedilir (2), uygulamanın AndroidManifest.xml dosyası ile yöntem kayıtları analiz edilir (3, 4) ve sonucunda rapor oluşturulur (5). Böylece uygulamanın tüm talep edilen izinleri kullanıp kullanmadığı belirlenir. Bu yöntemle incelenen uygulamaların manifest dosyasında talep ettiği izinler, statik kod analizi sonucunda tespit edilen izinler ve dinamik analiz sonucunda tespit edilen izinler belirlenmekte ve karşılaştırılmaktadır. Bu işlem sonucunda uygulamanın fazla izin talebinde bulunup bulunmadığı belirlenmektedir.

Geliştirilen yaklaşımın değerlendirilmesi maksadıyla Google Play mağazasında bulunan farklı kategorilerden her biri kendi kategorisinde ilk 500 içerisinde olan toplam 265 uygulama indirilmiştir. Bu uygulamalar hem gerçek mobil cihaz hem de Android 4.2.2 sürümünü çalıştıran Android emülatörü üzerinde analiz edilmiştir. Uygulamalarla ilgili çalışma zamanı bilgilerini ortaya çıkarabilmek maksadıyla uygulamalar Android Monkey test paketiyle 1500 farklı olay kullanılarak otomatik olarak çalıştırılmıştır. Yapılan işlemlerin sonucunda incelenen uygulamaların %87'sinin kullanmadığı halde fazladan izin talebinde bulunduğu belirlenmiştir.

An Android Behavior-Based Malware Detection Method using Machine Learning

Chang ve diğerleri yaptıkları çalışmada makine öğrenmesi tekniklerini kullanarak davranış tabanlı olarak kötücül yazılım tespit etmeyi sağlayan bir yöntem ortaya koymuşlardır [87]. Çalışma kapsamında DroidBox isimli Android uygulama kum havuzu aracını geliştirmişler ve bu araca mobil cihaz üzerinde çalışan uygulamalara anlamlı sırada tıklamayı sağlayan Robotium isimli bir görüntü tanımlama otomatik tetikleme programı eklemişlerdir. Çalışma kapsamında geliştirilen sistemin mimarisi Şekil 3.28a'da, tetikleme işleminin çalışma mantığı Şekil 3.28b'de gösterilmektedir.



a b

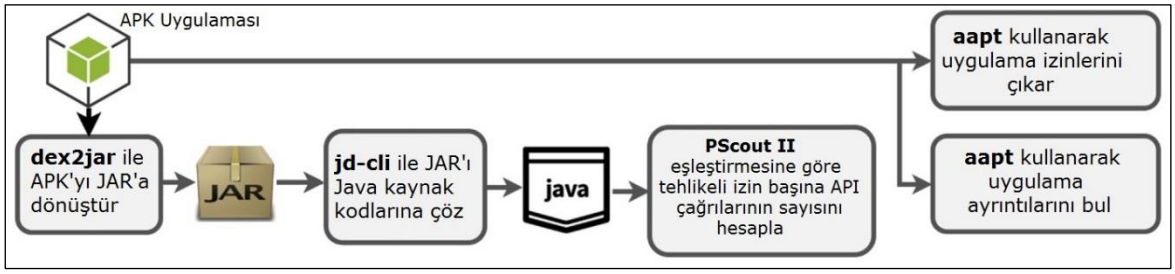
Şekil 3.28. a) Geliştirilen sistemin mimarisi b) Tetikleme işleminin çalışma mantığı [87]

Ön işleme adımında incelenecek olan uygulamanın AndroidManifest.xml dosyasından uygulama adı ve ana etkinlik ismi alınmaktadır. Ardından uygulamanın orijinal imzası silinmekte, tetikleyici ve uygulama aynı anahtar kullanılarak imzalanmakta ve bunlar Android emülatörüne yüklenmektedir. Veri izleyici bölümünde DroidBox statik ön kontrolü tamamladıktan sonra test çalıştırıcı çalışma kapsamında çalışma kapsamında geliştirilen Robotium test paketini çalıştırmaktadır. Hedef uygulama tetikleyici tarafından otomatik olarak yürütülmekte ve belirlenen olaylar gerçekleştirilmektedir. Robotium test paketi cihazın ekranı üzerinde gösterilen düğmeler, girdi/çıktı alanları vb. görsel öğeleri tanımlayabilmekte ve bunlara ilişkin olarak önceden belirlenmiş bir anlamlı sırada tıklama, yazma vb. işlemler yapabilmektedir.

Karar modelinde çeşitli makine öğrenmesi teknikleriyle testler yapılmıştır. Bu tekniklerde veri olarak uygulamanın çalışması sırasında elde edilen dosya okuma/yazma, kripto kullanımı, ağ işlemleri, başlatılan hizmetler, bilgi sızıntıları, telefon aramaları ve mesajları vb. bilgiler ile APK izinleri kullanılmaktadır. Çalışma sonucunda doğruluk oranının %97'ye kadar çıktığı ve yanlış pozitif oranının %4'ün altında olduğu bulunmuştur.

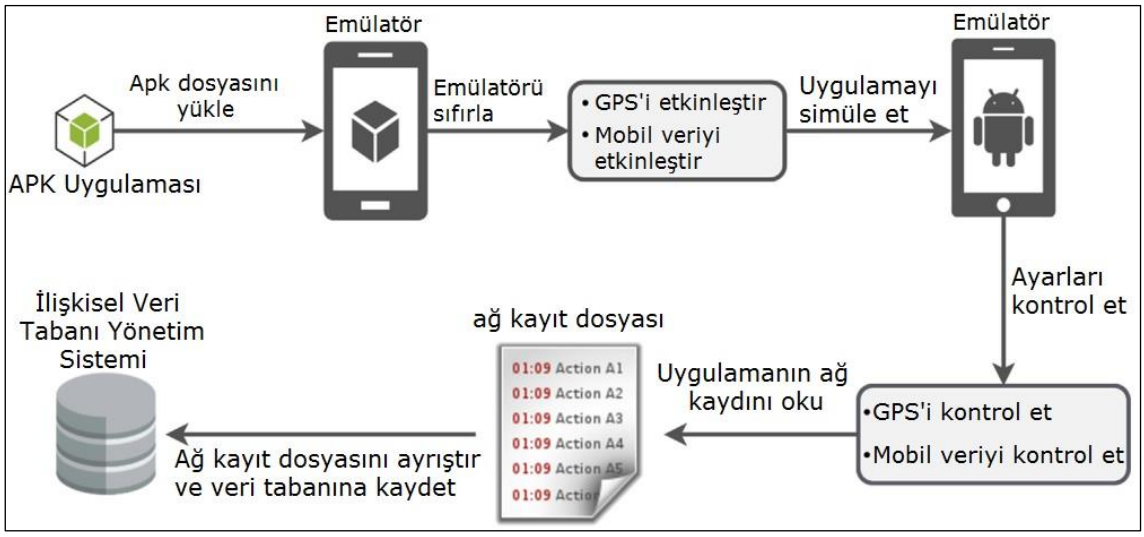
An in-depth analysis of Android malware using hybrid techniques

Kabakuş ve Doğru tarafından geliştirilen mad4a isimli yazılım statik analiz işleminde incelenen uygulamanın AndroidManifest.xml dosyası içerisinde talep edilen izinleri çıkarmakta ve çözülmüş uygulama kodu içerisinde bulunan API çağrılarını bulup bunları PScout II tarafından sağlanan ilgili izin gruplarıyla eşleştirmektedir [88]. Yazılımın statik analiz işleminin çalışma mantığı Şekil 3.29'da gösterilmiştir.



Şekil 3.29. mad4a yazılımının statik analiz işlemi [88]

Dinamik analiz bölümündeyse kullanıcı olaylarını simüle etmek maksadıyla Android SDK tarafından sağlanan monkey aracına dayanan MonkeyRunner kullanılarak emülatörde incelenen uygulama üzerinde 500 adet rastgele olay uygulanmış ve çalışma esnasında uygulamanın GPS ve ağ davranışları izlenmiştir. Yazılımın statik analiz işleminin çalışma mantığı Şekil 3.30'da gösterilmiştir.



Şekil 3.30. mad4a yazılımının dinamik analiz işlemi [88]

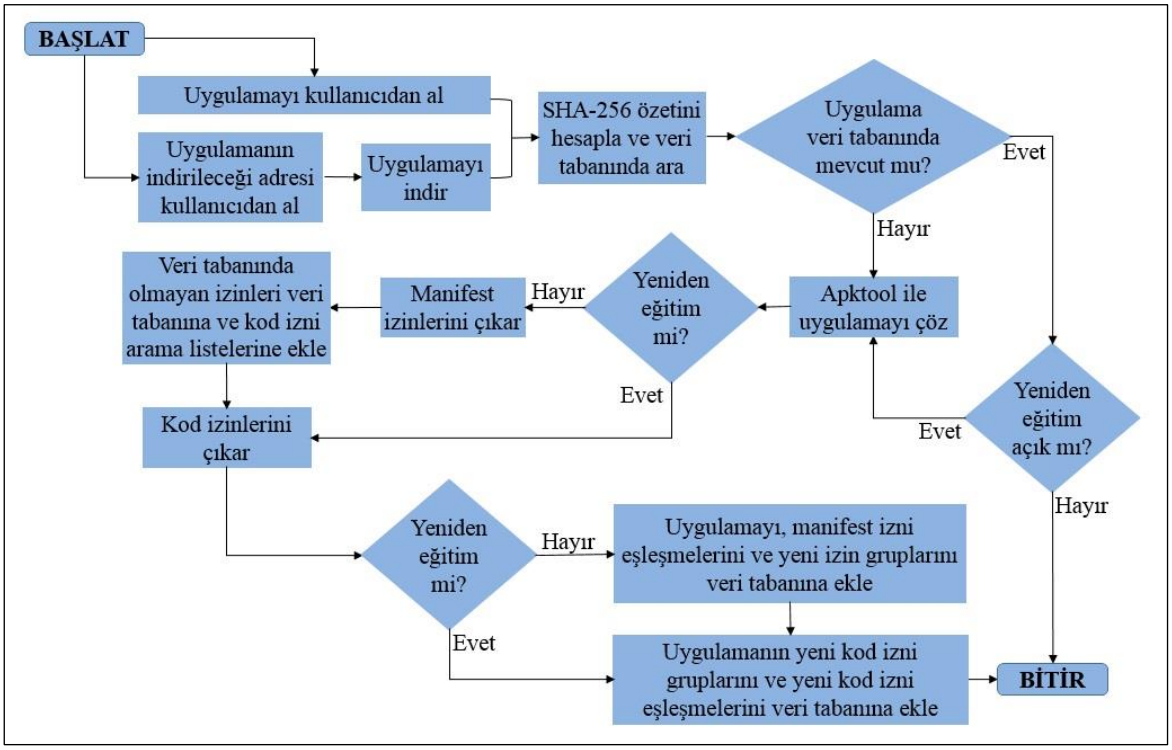
Çalışmanın değerlendirilmesi amacıyla Google Play uygulama mağazasından bilgileri alınan ve APKPure sitesinden indirilen 2999 iyicil uygulama ile Android Malware Genome Project, Drebin ve daha küçük boyutlu diğer kötücül uygulama veri kümelerinden toplam 2809 kötücül uygulama kullanılmıştır. Çalışma sonucunda incelenen iyicil uygulamalar ile kötücül uygulamaların izin talepleri, API çağrı sayıları, API çağrılarına göre izin kullanımları ve ağ davranışları arasındaki farklılıklar bulunmuştur.

4. ANDROID UYGULAMA ANALİZ SİSTEMİ (ANUAS)

Bu bölümde tez çalışması kapsamında geliştirilen ANUAS'ın sırasıyla çalışma mantığı, arayüzleri ve veri tabanı yapısı anlatılacaktır.

4.1. Çalışma Mantığı

ANUAS yüklenen ya da indirilen apk dosyalarını AndroidManifest.xml dosyaları içerisinde talep ettikleri izinler (bundan sonra manifest izinleri olarak adlandırılacaktır) ile çözülmüş kaynak kodları içesinde buluna izinleri (bundan sonra kod izinleri olarak adlandırılacaktır) incelemek ve bu izinlerden çıkarımlar yapmak suretiyle incelenen apk dosyalarının kötücül olup olmadığını tespit etmeye çalışan JSF ve Spring çatıları kullanılarak Java dilinde geliştirilmiş web tabanlı bir sistemdir. Sistemin çalışması eğitim ve analiz olmak üzere birbirine paralel yürütülebilen iki aşamadan oluşmaktadır. Sistemin eğitim aşamasının çalışma mantığı Şekil 4.1'de gösterilmiştir.



Şekil 4.1. Sistemin eğitim aşamasının çalışma mantığı

Bir uygulamayı analiz etmeden iyicil-kötücül durumunu doğrudan sisteme vermek suretiyle sistemin eğitimi sadece sistem yöneticileri tarafından yapılabilir. Eğitim aşamasında öncelikle incelenecek apk dosyasının SHA-256 özet değeri hesaplanır ve veri tabanında sorgulanarak uygulamanın daha önceden veri tabanına eklenmiş olup olmadığı kontrol edilir. Eğer incelenecek uygulama veri tabanında mevcutsa yeniden eğitim ayarı kontrol edilir. Yeniden eğitim ayarı kapalıysa uygulamanın veri tabanında mevcut olduğu ve uygulama bilgileri yöneticiye bildirilir ve başka işlem yapılmaz. Eğer uygulama mevcut değilse ya da yeniden eğitim ayarı açıksa öncelikle apktool programı kullanılarak uygulama çözülür.

Apktool programı APK formatındaki Android uygulamalarını dosya yapısını koruyarak çözebilme ve çözülmüş hallerinde bazı değişiklikler yapıldıktan sonra yeniden oluşturabilme yeteneklerine sahip bir tersine mühendislik aracıdır [89]. Apk dosyasını çözme işlemiyle çözülen apk dosyası içerisindeki AndroidManifest.xml dosyasını okunabilecek metin formatına getirerek orijinal haline dönüştürebilmektedir. Kaynak kodlarını ortaya çıkarmak içinse apk dosyası içerisindeki .dex dosyaları baksmali işlemine tabi tutulmakta ve assembly diline yakın olan smali formatında kodlar elde edilmektedir [90]. Apktool programının bir apk uygulamasını çözme adımları orijinal haliyle Şekil 4.2a'da ve Türkçe'ye çevrilmiş haliyle Şekil 4.2b'de gösterilmiştir.

```
D:\>java -jar apktool_2.4.0.jar d test.apk
I: Using Apktool 2.4.0 on test.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: 1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Baksmaling classes2.dex...
I: Baksmaling classes3.dex...
I: Baksmaling classes4.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
```

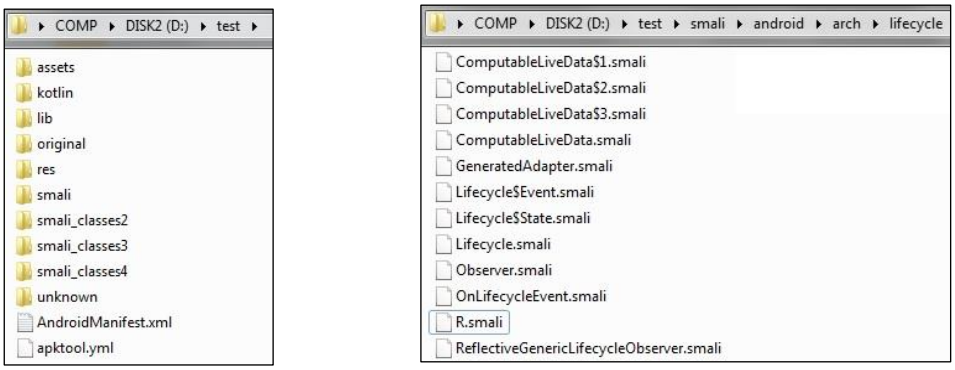
a

```
D:\>java -jar apktool_2.4.0.jar d test.apk
I: Test.apk üzerinde Apktool 2.4.0 kullanılıyor
I: Kaynaklar tablosu yükleniyor...
I: AndroidManifest.xml kaynaklar ile çözülüyor...
I: Kaynaklar tablosu dosyadan yükleniyor: 1.apk
I: Düzenli manifest paketi...
I: Dosya kaynakları çözülüyor...
I: */* XMLlerdeki değerler çözülüyor...
I: Classes.dex çözülüyor...
I: Classes2.dex çözülüyor...
I: Classes3.dex çözülüyor...
I: Classes4.dex çözülüyor...
I: Varlıklar ve kütüphaneler kopyalanıyor...
I: Bilinmeyen dosyalar kopyalanıyor...
I: Orijinal dosyalar kopyalanıyor...
```

b

Şekil 4.2. a) Apktool programının apk çözme adımları (Orijinal) b) Apktool programının apk çözme adımları (Türkçe)

Apktool programıyla çözülmüş örnek bir apk uygulaması klasörü Şekil 4.3a'da, çözülmüş örnek bir smali dosyaları klasörü Şekil 4.3b'de ve çözülmüş örnek bir smali kod dosyası Şekil 4.3c'de gösterilmiştir.



a b

FabricInitializer.smali

```
.class public Lio/fabric/unity/android/FabricInitializer;
.super Ljava/lang/Object;
.source "FabricInitializer.java"

# annotations
.annotation system Ldalvik/annotation/MemberClasses;
    value = {
        Lio/fabric/unity/android/FabricInitializer$Caller;
    }
.end annotation

# static fields
.field private static AUTOMATIC:Ljava/lang/String; = "Automatic"
.field private static MANUAL:Ljava/lang/String; = "Manual"
.field private static savedContext:Landroid/content/Context;

# direct methods
.method static constructor <clinit>()V
    .locals 5
    const-string v0, "Fabric|SafeDK: Execution> Lio/fabric/unity/android/FabricInitializer;-><clinit>()V"
    invoke-static {v0}, Lcom/safedk/android/utils/Logger;->d(Ljava/lang/String;)I
    const-string v0, "io.fabric.sdk.android"
    invoke-static {v0}, Lcom/safedk/android/internal/DexBridge;->isSDKEnabled(Ljava/lang/String;)Z
    move-result v1
```

Şekil 4.3. a) Apktool ile çözülmüş örnek bir apk uygulaması klasörü b) Apktool ile çözülmüş örnek bir smali dosyaları klasörü c) Apktool ile çözülmüş örnek bir smali kod dosyası

Uygulama apktool programı ile çözüldükten sonra çözülmüş apk klasöründe bulunan AndroidManifest.xml dosyasında talep edilen izinler okunarak uygulamanın manifest izinleri çıkarılır. Eğer izin adı veri tabanında mevcut değilse yeni izin olarak veri tabanına ve kod izni arama listelerine eklenir. Manifest izinleri sabit olduğundan ve zamanla değişmeyeceğinden manifest izinleriyle ilgili işlemler sadece uygulama veri tabanında zaten mevcut değilse yapılır.

Daha sonra aynı klasörde bulunan smali klasörlerindeki tüm .smali dosyaları taranarak izin adları aranır ve bulunan izinlerle uygulamanın kod izinleri çıkarılır. Dosyalardaki izin adları aranırken veri tabanının Izinler tablosunda bulunan izin adları ile eğitim aşamasında uygulamalar tarafından tanımlanan yeni izin adları kullanılır. Şekil 4.4'te smali kod dosyası içerisindeki örnek bir izin gösterilmiştir.

```
.method public hasEnrolledFingerprints()Z
    .locals 3
    .annotation build Landroid/support/annotation/RequiresPermission;
        value = "android.permission.USE_FINGERPRINT"
    .end annotation
```

Şekil 4.4. Smali kod dosyası içerisindeki örnek bir izin

Apk uygulamalarının apktool ile çözülmesi sonucunda çok sayıda smali dosyası oluşmakta ve bu dosyalarda tüm satırlarda tüm izin adlarını aramak çok uzun sürmektedir. Örnek olarak WhatsApp Messenger_v2.19.203.apk dosyasının apktool ile çözülmesi sonrasında smali klasörü içerisinde 677 alt klasör altında 12 253 adet .smali dosyası oluşmuştur ve bu dosyalarda boş satırlar hariç toplam 2 344 095 adet satır bulunmaktadır. Kod izni geçen her satırda . ve “ işaretleri bulunması gerekmektedir. Örnek uygulamanın smali dosyalarında bu iki işaretin de bulunduğu toplam 21 878 adet satır bulunmakta ve bu satırların sadece 144 adedinde bir izin adı geçmektedir. Android 9 (SDK 28) sürümünde ise 643 adet izin tanımlıdır. Uygulamaların tanımladıkları yeni izinlerin de eklenmesi halinde toplamda binlerce izin adı olmaktadır. Uygulama kodunda yer alan ve . ile “ işaretlerinin bulunduğu binlerce satırın her birinde binlerce izin adını aramak çok fazla vakit almaktadır. Bu nedenle ANUAS’ın kod izinlerini arama algoritmasında anahtarlama kullanılmıştır.

Bu yönteme göre Android izinlerinin baştan ilk noktaya kadar olan kısımları ile birincil anahtarlar, baştan ikinci noktaya kadar olan kısımları ile ikincil anahtarlar oluşturulmaktadır. Örneğin “android.permission.INTERNET” izni için birincil anahtar “android", ikincil anahtar “android.permission” şeklindedir. Tarama yapılırken önce . ve “ işaretlerinin olduğu satırlar aranmakta, satır bulunması halinde satırda birincil anahtarlar aranmakta, birincil anahtar bulunması halinde bulunan birincil anahtarın ikincil anahtarları satırda aranmakta, ikinci anahtar bulunması halinde bulunan ikincil anahtarın tüm izin adları satırda aranmakta ve izin adı bulunması durumunda bulunan izin adı eğer daha önce eklenmemişse kod izinleri kümesine eklenmektedir. ANUAS’ta bir uygulamanın .smali dosyaları aynı anda 50 iş parçacığı oluşturularak eş zamanlı olarak bu yönteme göre taranır ve çok daha kısa sürede uygulamanın kod izinleri bulunur. Örnek olarak Android 9 (SDK 28) sürümünde tanımlı 643 adet izin adından oluşturulan birincil ve ikincil anahtarlar ve bu anahtarlara bağlı izin sayıları Çizelge 4.1’de sunulmuştur.

Çizelge 4.1. Android 9 (SDK 28) sürümü izinlerinin arama anahtarları

Birincil Anahtarlar	İkincil Anahtarlar	İzin Sayısı
android	android.intent	1
	android.permission	461
	android.server	1
com	com.android	22
	com.breel	1
	com.google	157

ANUAS'ın kod izinlerini arama görevi kodu Şekil 4.5'te gösterilmiştir.

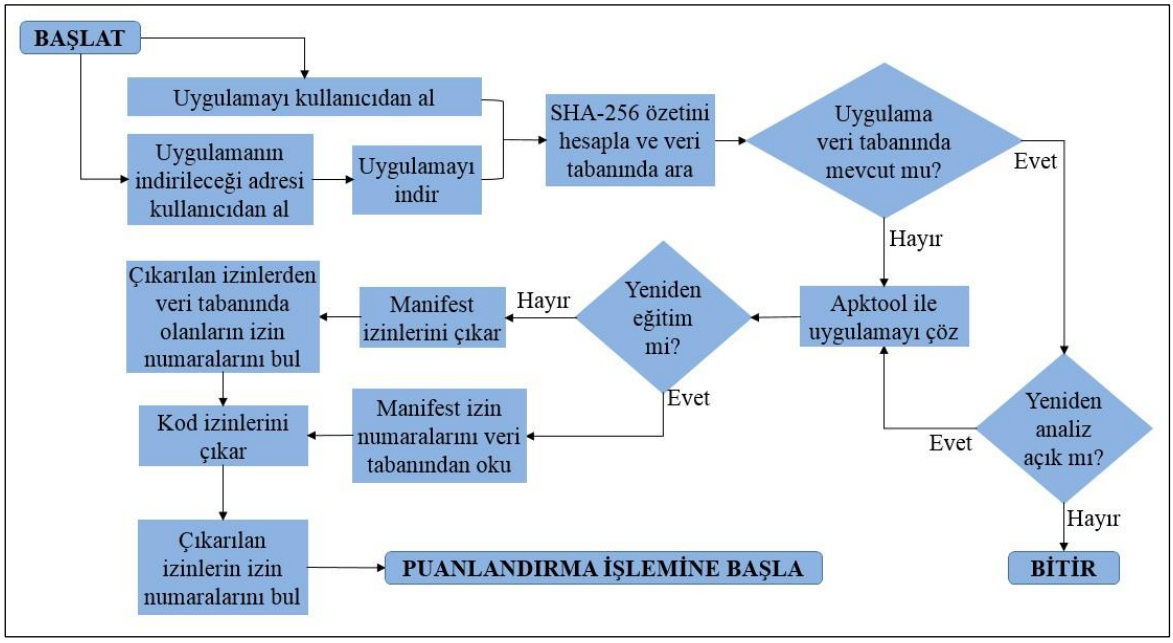
```
while ((dosyaSatiri = tamponluOkuyucu.readLine()) != null) {
    if (dosyaSatiri.contains(".") && dosyaSatiri.contains("\"") ) {
        tumForDonguleri:
        for(String geciciBirincilAnahtar : DosyaIslemci.birincilAnahtarlar)
            if(dosyaSatiri.contains(geciciBirincilAnahtar + ".")) {
                geciciIkincilAnahtarlarKumesi = DosyaIslemci.ikincilAnahtarlar.
                                                get(geciciBirincilAnahtar);
                for(String geciciIkincilAnahtar : geciciIkincilAnahtarlarKumesi)
                    if(dosyaSatiri.contains(geciciIkincilAnahtar + ".")) {
                        geciciIkincilAnahtarIzinAdlariListesi = DosyaIslemci.tumIzinAdlari.
                                                                get(geciciBirincilAnahtar).
                                                                get(geciciIkincilAnahtar);
                        for(String izinAdi : geciciIkincilAnahtarIzinAdlariListesi)
                            if(dosyaSatiri.contains(izinAdi)) {
                                DosyaIslemci.kodIzinlerineEkle(izinAdi);
                                break tumForDonguleri;
                            }
                    }
            }
    }
}
```

Şekil 4.5. Kod izinlerini arama görevi kodu

Android SDK sürümlerinin değişmesine ya da uygulamaların yeni izinler tanımlamasına bağlı olarak aranması gereken izinler değişebilir. Bu nedenle yeniden eğitim durumunda sadece uygulamanın veri tabanında mevcut olanların dışındaki yeni kod izni eşleşmeleri ve yeni izin grupları veri tabanına eklenir.

Tüm manifest ve kod izinleri çıkarıldıktan sonra uygulamayı veri tabanına ekleme süreci başlar. Bu süreçte uygulama bilgileri ve uygulama-izin eşleşmeleri veri tabanına eklenir. Ardından manifest ve kod izinlerinden ikili ve üçlü izin grupları oluşturulur, bunlar veri tabanında sorgulanır, veri tabanında bulunmayan yeni izin grupları veri tabanına eklenir.
Sistemin analiz aşamasında eğitim aşaması ile aynı şekilde öncelikle incelenecek apk dosyasının SHA-256 özet değeri hesaplanır ve veri tabanında sorgulanarak uygulamanın

daha önceden veri tabanına eklenmiş olup olmadığı kontrol edilir. Eğer incelenecek uygulama veri tabanında mevcutsa yeniden analiz ayarı kontrol edilir. Yeniden analiz ayarı kapalıysa uygulamanın veri tabanında mevcut olduğu ve uygulama bilgileri kullanıcıya bildirilir ve başka işlem yapılmaz. Eğer uygulama mevcut değilse ya da yeniden analiz ayarı açıksa öncelikle apktool programı kullanılarak uygulama çözülür. Uygulama çözüldükten sonra çözülmüş apk klasöründe bulunan AndroidManifest.xml dosyasında talep edilen izinler okunarak uygulamanın manifest izinleri çıkarılır, bunlardan veri tabanına eklenmiş olanların izin numaraları bulunur. Uygulama yeniden analiz ediliyorsa manifest izin numaraları veri tabanından alınır. Ardından aynı klasörde bulunan smali klasörlerindeki tüm .smali dosyaları taranarak izin adları aranır, bulunan izinlerle uygulamanın kod izinleri çıkarılır ve bu izinlerin izin numaraları bulunur. Sistemde analiz edilecek izin numaralarının belirlenmesi işleminin çalışma mantığı Şekil 4.6'da gösterilmiştir.



Şekil 4.6. Analiz edilecek izin numaralarının belirlenmesi işleminin çalışma mantığı

İncelenen uygulamanın izinlerini belirleme ve izin gruplarını oluşturma işlemi bittikten sonra puanlandırma süreci başlamaktadır. ANUAS'ta incelenen her uygulama için,

- Tekil manifest izinleri puanı,
- Tekil kod izinleri puanı,
- İkili manifest izin grupları puanı,
- İkili kod izin grupları puanı,

- Üçlü manifest izin grupları puanı ve
- Üçlü kod izin grupları puanı olmak üzere altı farklı risk puanı hesaplanmaktadır.

Puanlandırma için öncelikle incelenen uygulamanın talep ettiği veya kod içerisinde kullandığı her bir iznin ya da izin grubunun kötücül uygulamalarda bulunma oranı (KO) ve iyicil uygulamalarda bulunma oranı (İO) belirlenir. Örnek olarak manifest izinleri puanı hesaplanırken bir iznin kötücül uygulamalarda bulunma oranı, bu izni talep eden kötücül uygulama sayısının veri tabanındaki tüm kötücül uygulamaların sayısına bölümüyle elde edilir (Eş. 4.1). Aynı şekilde iyicil uygulamalarda bulunma oranı, bu izni talep eden iyicil uygulama sayısının veri tabanındaki tüm iyicil uygulamaların sayısına bölümüyle elde edilir (Eş. 4.2).

KO (İzin) = Talep eden kötücül uygulama sayısı / Toplam kötücül uygulama sayısı (4.1)

İO (İzin) = Talep eden iyicil uygulama sayısı / Toplam iyicil uygulama sayısı (4.2)

Bir iznin kötücül ve iyicil uygulamalarda bulunma oranları belirlendikten sonra bu iznin risk puanı (RP) şu eşitlikle hesaplanır:

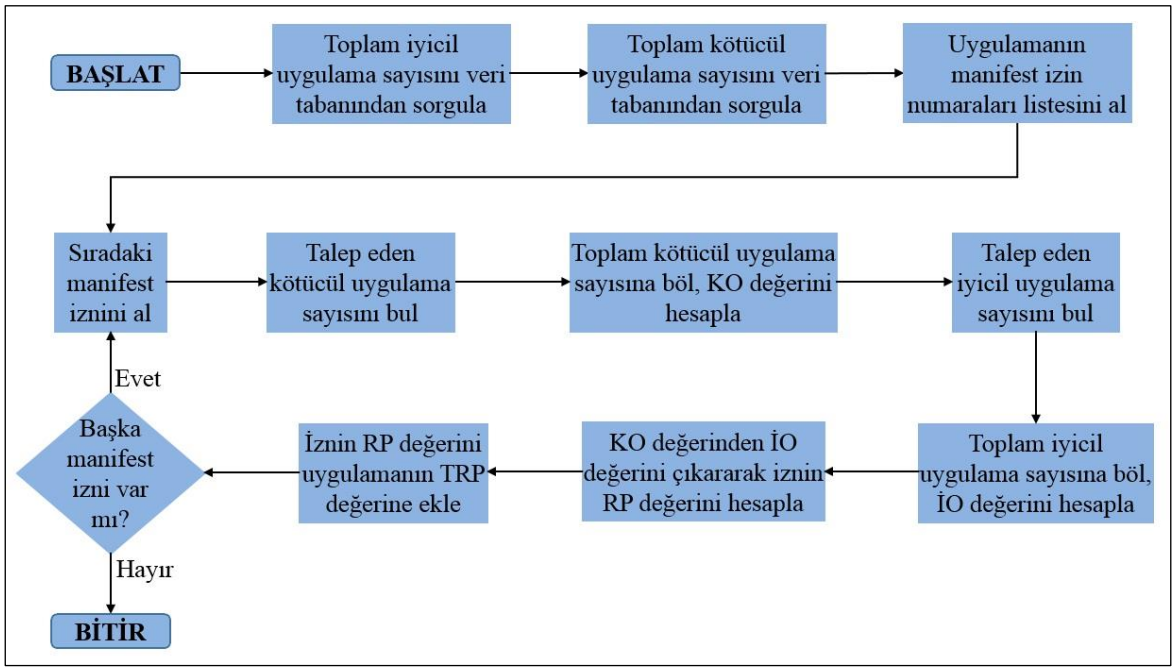
RP (İzin) = KO (İzin) - İO (İzin) (4.3)

Bu eşitliğe göre herhangi bir iznin kötücül uygulamalarda bulunma oranı arttıkça risk puanı artmakta, iyicil uygulamalarda bulunma oranı arttıkça risk puanı azalmaktadır. İncelenen uygulamanın tekil manifest izinleri puanı hesaplanırken talep ettiği tüm izinler için KO, İO ve RP değerleri bulunur, ardından uygulamanın toplam risk puanı (TRP) şu eşitlikle hesaplanır:

$$\text{TRP (Uygulama)} = \sum_{i=1}^{n} \text{RP(İzin(i))} \quad (4.4)$$

Tekil manifest izinleri puanı hesaplanırken bu eşitlikte n uygulamanın talep ettiği izin sayısını göstermektedir. Eş. 4.4'e göre tekil manifest izinleri puanını bulmak için önce talep edilen tüm izinlerin risk puanları hesaplanır, ardından tüm risk puanları toplanarak

uygulamanın toplam risk puanı hesaplanır. Sistemde bir uygulamanın tekil manifest izinleri puanının hesaplanması işleminin çalışma mantığı Şekil 4.7'de gösterilmiştir.

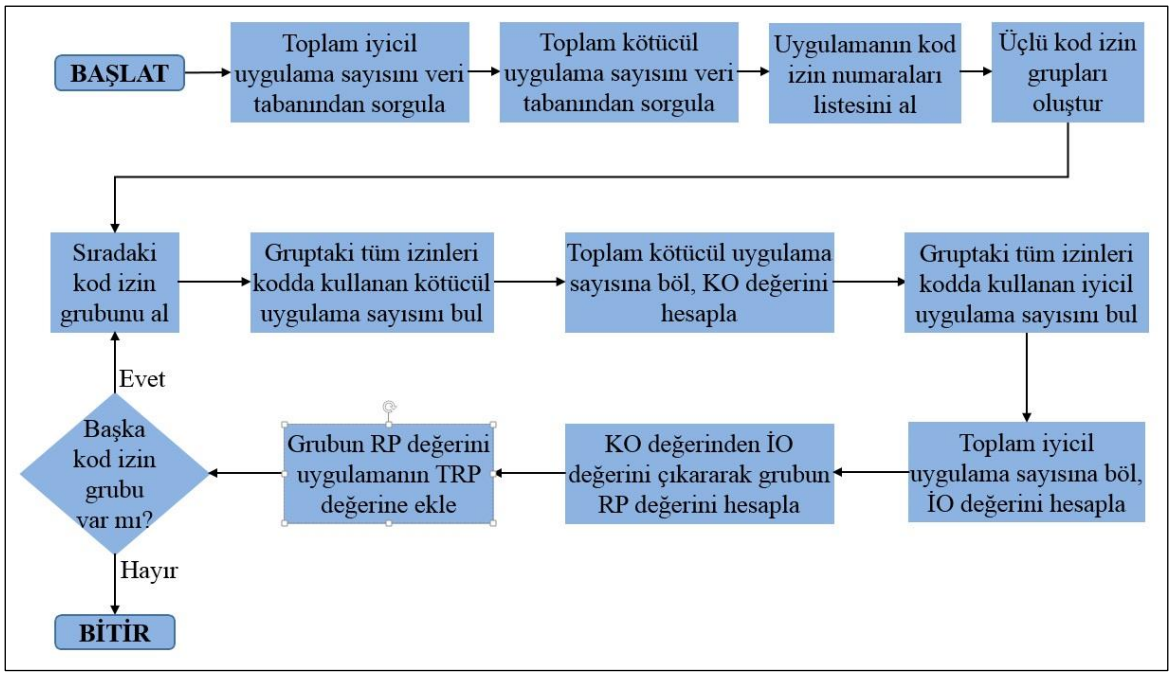


Şekil 4.7. Tekil manifest izinleri puanının hesaplanması işleminin çalışma mantığı

Kod izinleri puanları hesaplanırken uygulamanın kod içerisinde kullandığı bir iznin KO değeri, izni kod içerisinde kullanan kötücül uygulama sayısının veri tabanındaki tüm kötücül uygulamaların sayısına bölümüyle elde edilir. Aynı şekilde iznin İO değeri, izni kod içerisinde kullanan iyicil uygulama sayısının veri tabanındaki tüm iyicil uygulamaların sayısına bölümüyle elde edilir.

İkili ve üçlü izin grupları hesaplanırken ise izin yerine bu izinlerden oluşturulan izin gruplarının talep edilme/kod içerisinde kullanılma oranları dikkate alınır. Örneğin 6 adet manifest izni olan bir uygulamanın 15 (6*5 / 2) adet ikili manifest izin grubu vardır. Uygulamanın ikili manifest izin grupları puanını bulmak için öncelikle 15 adet ikili manifest izin grubunun risk puanları hesaplanır. Her bir izin grubu için gruptaki iki izni de talep eden kötücül ve iyicil uygulamaların sayısı dikkate alınır. Ardından izin gruplarının risk puanları toplanarak uygulamanın ikili manifest izin grupları puanı bulunur. Başka bir örnek olarak 30 adet kod izni olan bir uygulamanın 4060 (30*29*28 / 6) adet üçlü kod izin grubu vardır. Uygulamanın üçlü kod izin grupları puanını bulmak için öncelikle 4060 adet üçlü kod izin grubunun risk puanları hesaplanır. Her bir izin grubu için gruptaki üç izni de kod içerisinde

kullanan kötücül ve iyicil uygulamaların sayısı dikkate alınır. Ardından izin gruplarının risk puanları toplanarak uygulamanın üçlü kod izin grupları puanı bulunur. Sistemde bir uygulamanın üçlü kod izin grupları puanının hesaplanması işleminin çalışma mantığı Şekil 4.8'de gösterilmiştir.



Şekil 4.8. Üçlü kod izin grupları puanının hesaplanması işleminin çalışma mantığı

ANUAS'ta incelenen uygulamanın talep ettiği veya kod içerisinde kullandığı her bir iznin ya da izin grubunun kötücül ve iyicil uygulamalarda bulunma oranları HashMap eşleşme tablosu yapısı kullanılarak bellekten sorgulanmaktadır. HashMap<anahtar, değer> yapısı Java dilinde en popüler Map arayüzü uygulamalarından biri olup O(1) yani sabit zamanda istenen değere ulaşabilmeyi sağlamaktadır [91]. Sistemin geliştirilmesi aşamasında en hızlı yöntemin belirlenmesi maksadıyla bellekten ve veri tabanından sorgulama arasındaki performans farkı incelenmiştir. Bunun için önce veri tabanında UcluIzinGruplari tablosuna test amacıyla yerleştirilen 5 000 000 adet üçlü izin grubunun grup elemanları ve ManIyiSayisi değerleri toplu olarak sorgulanarak bir eşleşme tablosu içerisine konmuştur. Bu işlem ortalama olarak yaklaşık 41 saniyesi veri tabanından sorgulama, 10 saniyesi sonuç kümesini eşleşme tablosu içerisine yerleştirme olmak üzere toplamda 51 saniyede tamamlanmıştır. Ardından rastgele 10 000 adet üçlü izin grubu belirlenmiş ve bu izin gruplarının ManIyiSayisi değerleri tek tek hem bellekte bulunan eşleşme tablosundan hem de veri tabanından sorgulanmıştır. Yapılan test aynı eşleşme tablosu kullanılarak ve her

seferinde yeniden rastgele 10 000 adet üçlü izin grubu seçilerek 20 defa tekrarlanmıştır. Yapılan testlerin sonucu Çizelge 4.2'de gösterilmiştir.

Çizelge 4.2. Sorgulama testleri sonuçları

Test No	Eşleşme Tablosundan Sorgulama Süresi	Veri Tabanından Sorgulama Süresi
1	9 ms	162853 ms
2	6 ms	153349 ms
3	5 ms	131440 ms
4	5 ms	119080 ms
5	5 ms	109164 ms
6	5 ms	92494 ms
7	5 ms	93904 ms
8	5 ms	80525 ms
9	5 ms	70680 ms
10	5 ms	60321 ms
11	5 ms	54418 ms
12	5 ms	49241 ms
13	6 ms	52610 ms
14	6 ms	40593 ms
15	5 ms	37782 ms
16	5 ms	35356 ms
17	5 ms	31483 ms
18	7 ms	32388 ms
19	6 ms	28324 ms
20	6 ms	27858 ms
Ortalama	6 ms	73193 ms

Çizelgeden de görülebileceği üzere izin ya da izin gruplarının kötücül ve iyicil uygulamalarda bulunma oranlarını veri tabanından sorgulama işlemi her testte daha kısa sürede tamamlanmaktadır. Ancak yine de bellekteki eşleşme tablosundan sorgulama ile veri tabanından sorgulama arasında oldukça büyük bir hız farkı bulunmaktadır. Bu nedenle ANUAS'ta bellekteki eşleşme tablosundan sorgulama yöntemi tercih edilmiştir. Bu yönteme göre veri tabanındaki izinler, ikili izin grupları ve üçlü izin grupları tablolarının ManIyiSayisi, ManKotuSayisi, KodIyiSayisi ve KodKotuSayisi sütunlarından her biri için değeri sıfırdan büyük olan satırları kapsayan birer eşleşme tablosu oluşturulmuştur. Oluşturulan tablolar ve bunları oluşturma işleminin kodu Şekil 4.9'da gösterilmiştir.

```
private static void izinSayilariEslesmeTablolariniOlustur() {
    long tekilIzinTablolariniOlusturmaBaslangicZamani = System.currentTimeMillis();
    tekilIzinManIyiSayilari     = VeriTabaniIslemci.tureGoreIzinlerinTalepEdilmeSayilariniOlustur("IYICIL");
    tekilIzinManKotuSayilari    = VeriTabaniIslemci.tureGoreIzinlerinTalepEdilmeSayilariniOlustur("KOTUCUL");
    tekilIzinKodIyiSayilari     = VeriTabaniIslemci.tureGoreIzinlerinKullanilmaSayilariniOlustur("IYICIL");
    tekilIzinKodKotuSayilari    = VeriTabaniIslemci.tureGoreIzinlerinKullanilmaSayilariniOlustur("KOTUCUL");
    System.out.println("TEKİL İZİN EŞLEŞME TABLOLARINI OLUŞTURMA SÜRESİ: "
                    + (System.currentTimeMillis() - tekilIzinTablolariniOlusturmaBaslangicZamani) + " ms");

    long ikiliIzinGrubuTablolariniOlusturmaBaslangicZamani = System.currentTimeMillis();
    ikiliIzinGrubuManIyiSayilari    = VeriTabaniIslemci.tureGoreIkiliIzinGruplarininTalepEdilmeSayilariniOlustur("IYICIL");
    ikiliIzinGrubuManKotuSayilari   = VeriTabaniIslemci.tureGoreIkiliIzinGruplarininTalepEdilmeSayilariniOlustur("KOTUCUL");
    ikiliIzinGrubuKodIyiSayilari    = VeriTabaniIslemci.tureGoreIkiliIzinGruplarininKullanilmaSayilariniOlustur("IYICIL");
    ikiliIzinGrubuKodKotuSayilari   = VeriTabaniIslemci.tureGoreIkiliIzinGruplarininKullanilmaSayilariniOlustur("KOTUCUL");
    System.out.println("İKİLİ İZİN GRUBU EŞLEŞME TABLOLARINI OLUŞTURMA SÜRESİ: "
                    + (System.currentTimeMillis() - ikiliIzinGrubuTablolariniOlusturmaBaslangicZamani) + " ms");

    long ucluIzinGrubuTablolariniOlusturmaBaslangicZamani = System.currentTimeMillis();
    ucluIzinGrubuManIyiSayilari     = VeriTabaniIslemci.tureGoreUcluIzinGruplarininTalepEdilmeSayilariniOlustur("IYICIL");
    ucluIzinGrubuManKotuSayilari    = VeriTabaniIslemci.tureGoreUcluIzinGruplarininTalepEdilmeSayilariniOlustur("KOTUCUL");
    ucluIzinGrubuKodIyiSayilari     = VeriTabaniIslemci.tureGoreUcluIzinGruplarininKullanilmaSayilariniOlustur("IYICIL");
    ucluIzinGrubuKodKotuSayilari    = VeriTabaniIslemci.tureGoreUcluIzinGruplarininKullanilmaSayilariniOlustur("KOTUCUL");
    System.out.println("ÜÇLÜ İZİN GRUBU EŞLEŞME TABLOLARINI OLUŞTURMA SÜRESİ: "
                    + (System.currentTimeMillis() - ucluIzinGrubuTablolariniOlusturmaBaslangicZamani) + " ms");
}
```

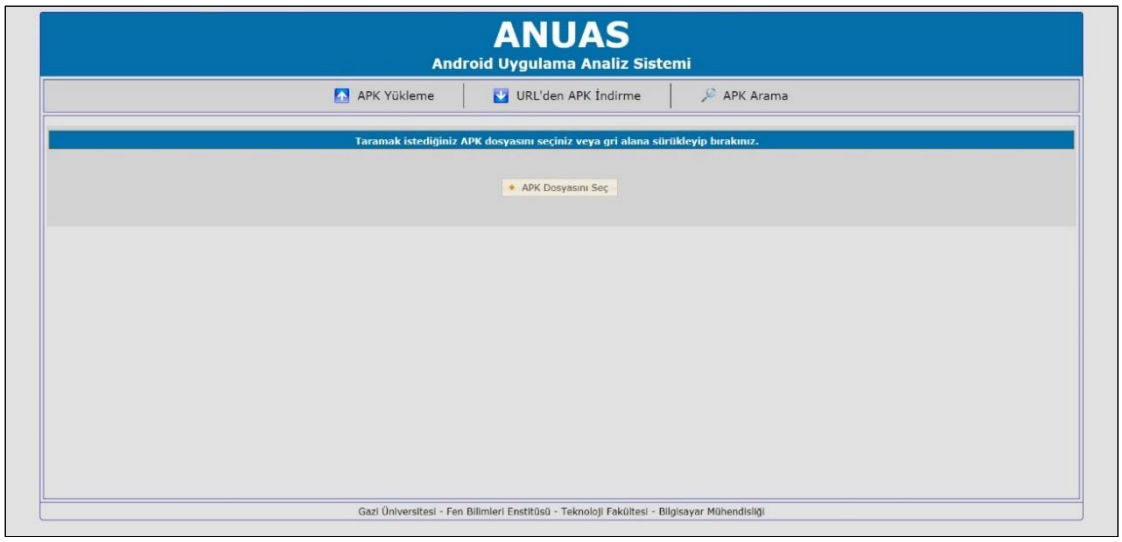
Şekil 4.9. Oluşturulan eşleşme tabloları

Tekil izinlere ilişkin eşleşme tablolarında anahtar olarak izin numarası, ikili izin gruplarına ilişkin eşleşme tablolarında anahtar olarak "İzin1 Numarası,İzin2 Numarası", üçlü izin gruplarına ilişkin eşleşme tablolarında ise anahtar olarak "İzin1 Numarası,İzin2 Numarası,İzin3 Numarası" kullanılmıştır. Tüm tablolarda değer olarak ise iznin/izin grubunun kötücül/iyicil uygulamalarda talep edilme/kod içerisinde kullanılma sayıları kullanılmıştır.

İncelenen uygulamaya ilişkin altı farklı puan hesaplandıktan sonra daha önceden belirlenen eşik değerler ile karşılaştırılarak uygulamanın kötücül olup olmadığı değerlendirilir. Analiz edilen uygulamaların veri tabanına kaydedilmesi ayarı açıksa uygulama bilgileri, uygulamanın tanımladığı yeni izinler, talep ettiği ya da kod içerisinde kullandığı yeni izin grupları ve uygulama-izin eşleşmeleri veri tabanına eklenir.

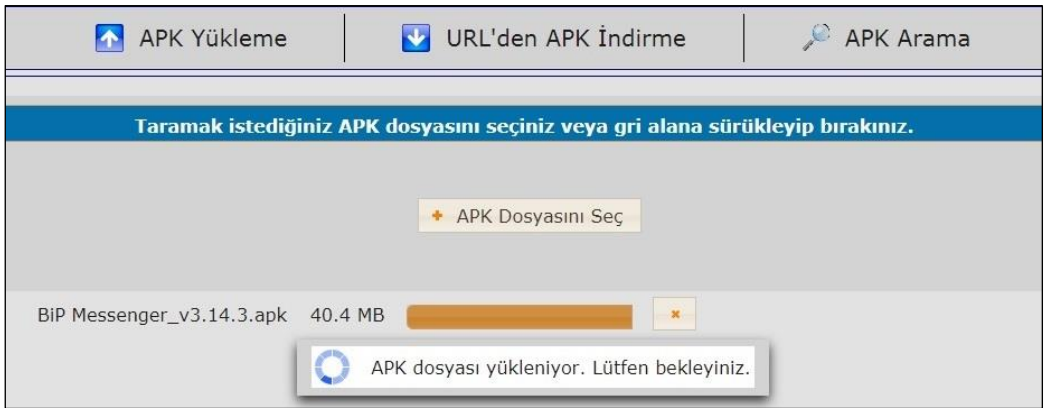
4.2. Sistem Arayüzleri

ANUAS'ın arayüzlerini geliştirmek için PrimeFaces JSF kütüphanesi kullanılmıştır. PrimeFaces Java ekosisteminde en popüler arayüz kütüphanelerinden biri olup Lufthansa, BMW, Ford, Cisco, Siemens, Symantec vb. gibi dünyaca ünlü pek çok şirket tarafından kullanılmaktadır [92]. ANUAS'ta analiz edilecek APK dosyası kullanıcı tarafından sisteme yüklenebilir ya da sisteme dosyanın indirilebileceği bir web bağlantısı verilerek APK dosyasının sistem tarafından indirilmesi sağlanabilir. APK yükleme arayüzü Şekil 4.10'da sunulmuştur.

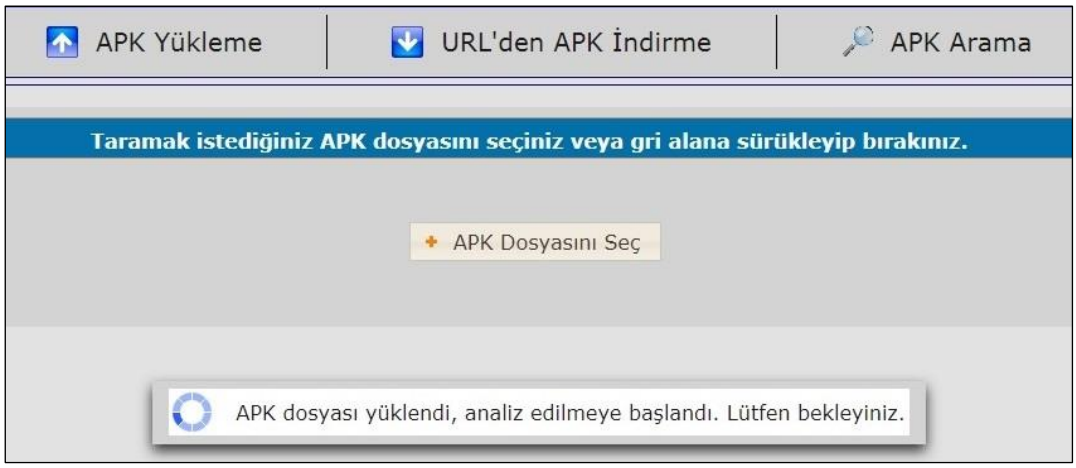


Şekil 4.10. APK dosyası yükleme arayüzü

APK yükleme arayüzünde analiz edilecek APK dosyasının seçilebileceği bir dosya seçme düğmesi mevcuttur. Kullanıcılar tarafından bu düğme kullanılarak aynı anda tek bir APK dosyası seçilebilmektedir. Seçilen dosya otomatik olarak sisteme yüklenir ve analiz edilmeye başlanır. APK yükleme arayüzü aynı zamanda sürükle-bırak özelliğini de desteklemektedir. Dosya seçme düğmesinin bulunduğu koyu gri alana sürüklenip bırakılan APK dosyaları otomatik olarak sisteme yüklenir ve analiz edilmeye başlanır. Ayrıca sadece APK dosyalarının yüklenebilmesi amacıyla dosya uzantısı kontrolü ve sadece 150 MB'a kadar olan dosyaların yüklenebilmesi amacıyla dosya büyüklüğü kontrolü de mevcuttur. Dosyanın yüklenmesi ya da analiz edilmesi sırasında kullanıcıların sayfayı değiştirmelerinin engellenmesi amacıyla tüm sayfa BlockUI gizli penceresi ile kaplanmakta ve pencere üzerinde Şekil 4.11 ve Şekil 4.12'de gösterildiği gibi durum kullanıcıya bildirilmektedir.

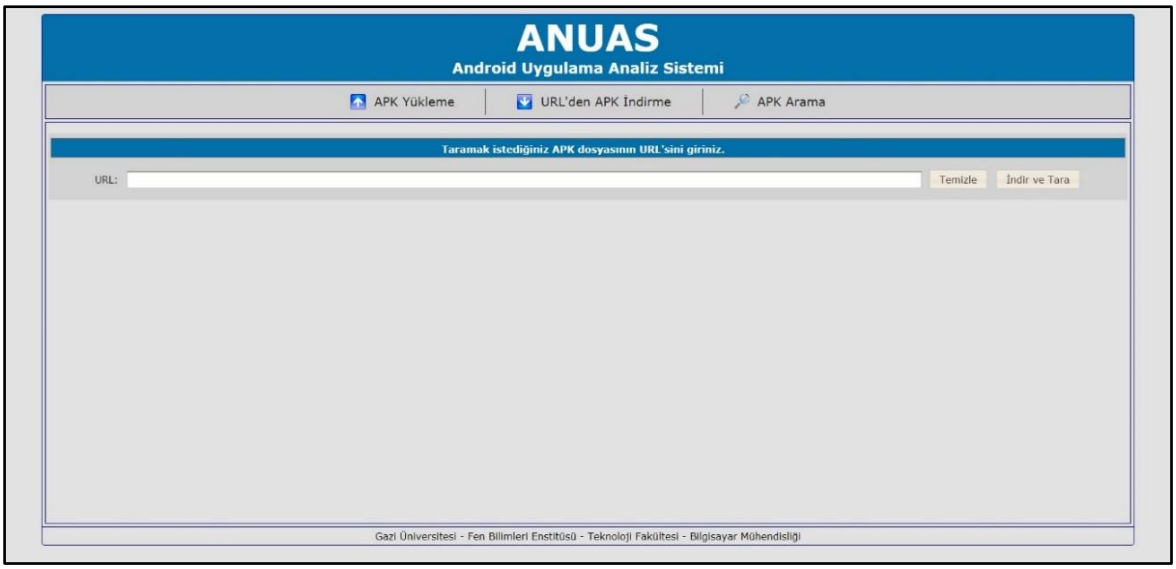


Şekil 4.11. APK dosyası yükleniyor bildirimi



Şekil 4.12. APK dosyası yüklendi bildirimi

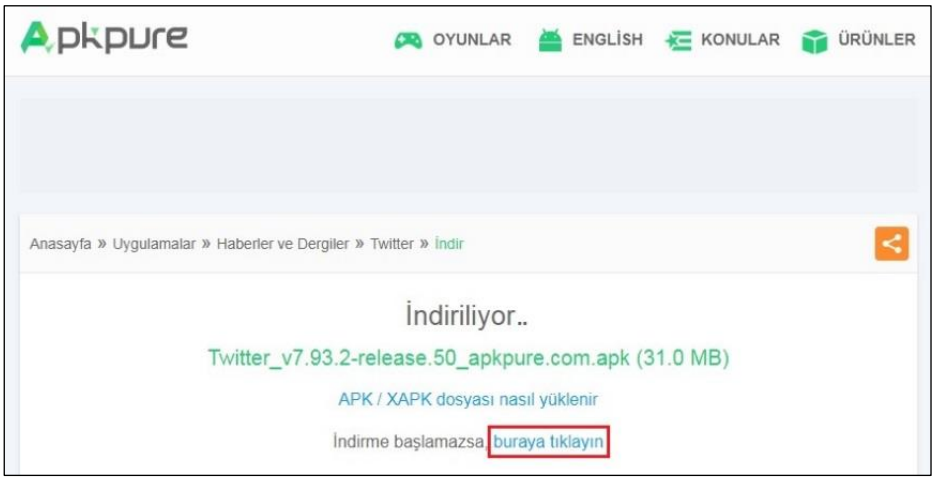
URL'den APK indirme arayüzünde analiz edilecek APK dosyasının indirilebileceği web bağlantısının yazılacağı bir girdi alanı ile indirme ve analiz işlemini başlatma düğmesi mevcuttur. Sistem kullanıcıları tarafından bu düğme kullanılarak aynı anda tek bir apk dosyasının indirilme ve analiz işlemi başlatılabilmektedir. URL'den APK indirme arayüzü Şekil 4.13'te sunulmuştur.



Şekil 4.13. URL'den APK indirme arayüzü

URL alanına analiz edilecek APK dosyasının indirilebileceği web bağlantısının girilmesi gerekmektedir. Kullanıcılar bu web bağlantısını çeşitli APK indirme sitelerinden bulabilmektedir. Örneğin Şekil 4.14'te gösterildiği gibi APKPure sitesinde APK indirme sayfasında bulunan *buraya tıklayın* yazısının üzerine gelinip bağlantı kopyalanabilir, bu

yolla indirilen APK dosyasının indirme bağlantısı elde edilebilir ve ardından bu bağlantı ANUAS'ın URL'den APK indirme arayüzünde bulunan URL alanına girilebilir.

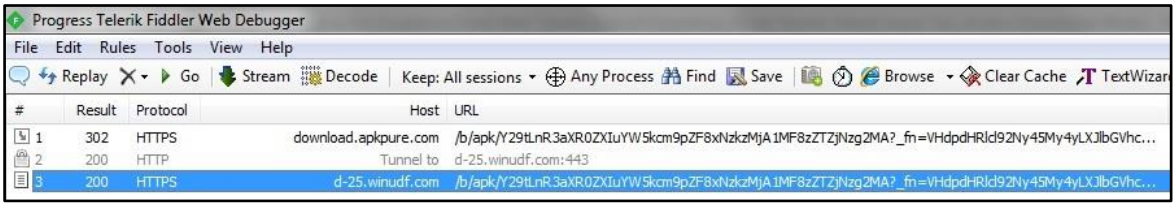


Şekil 4.14. APK indirme sayfasında bulunan indirme bağlantısı

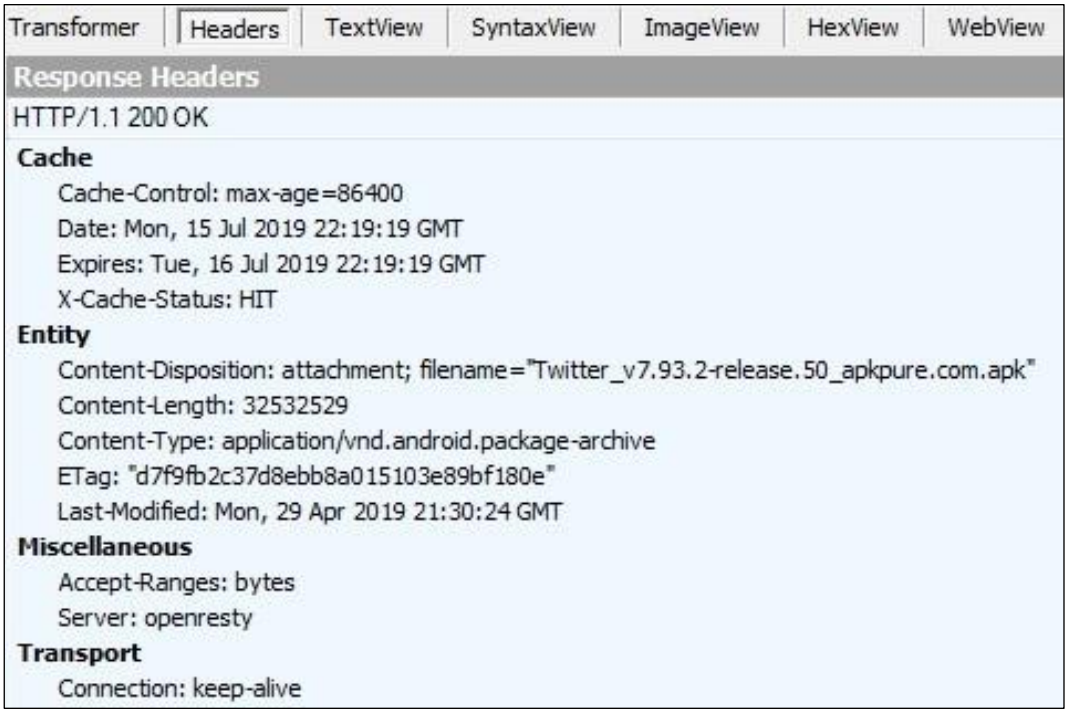
APKPure sitesinde bulunan indirme bağlantıları indirilen APK dosyasının adının içermemektedir. Örneğin *Twitter_v7.93.2-release.50_apkpure.com.apk* dosyasının geçici indirme bağlantısı şu şekildedir:

https://download.apkpure.com/b/apk/Y29tLnR3aXR0ZXIuYW5kcm9pZF8xNzkzMjA1MF8zZTZjNzg2MA?_fn=VHdpdHRlcl92Ny45My4yLXJlbGVhc2UuNTBfYXBrcHVyZS5jb20uYXBr&k=70550b97a9565cf9e16c9438e7c4227f5d2f8c1b&as=34e35c16b7f35d0721523a026e94ccca5d2ce993&_p=Y29tLnR3aXR0ZXIuYW5kcm9pZA&c=1%7CNEWS_AND_MAGAZINES%7CZGV2PVR3aXR0ZXIlMkMlMjBJbmMuJnQ9YXBrJnM9MzI1MzI1Mjkmdm49Ny45My4yLXJlbGVhc2UuNTAmdmM9MTc5MzIwNTA&hot=1

Bu bağlantıya tıklandığında Şekil 4.15'te gösterildiği gibi istek yönlendirmesi yapılmakta ve Şekil 4.16'da gösterildiği gibi yönlendirilen siteden gelen cevabın içerisinde bulunan *Content-Disposition* başlık alanının *filename* kısmında indirilen dosyanın adı yazmaktadır. ANUAS verilen URL adresinden yapılan yönlendirmeleri takip edebilmekte ve son ulaşılan web sayfasından gelen cevabın içerisinden indirilen APK dosyasının adını alabilmektedir.

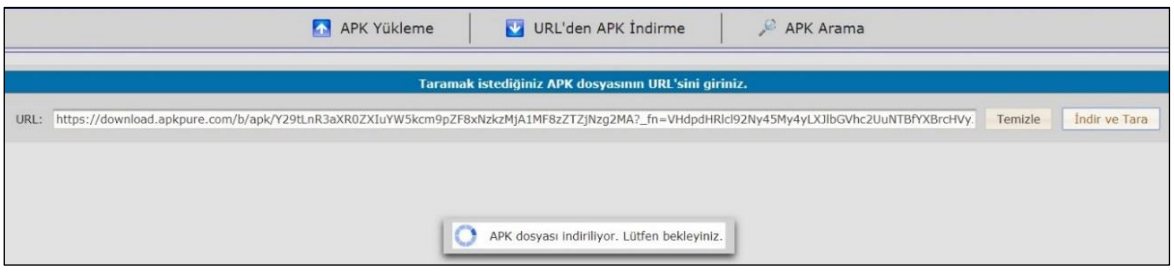


Şekil 4.15. APKPure web sitesinden APK dosyası indirme trafiği

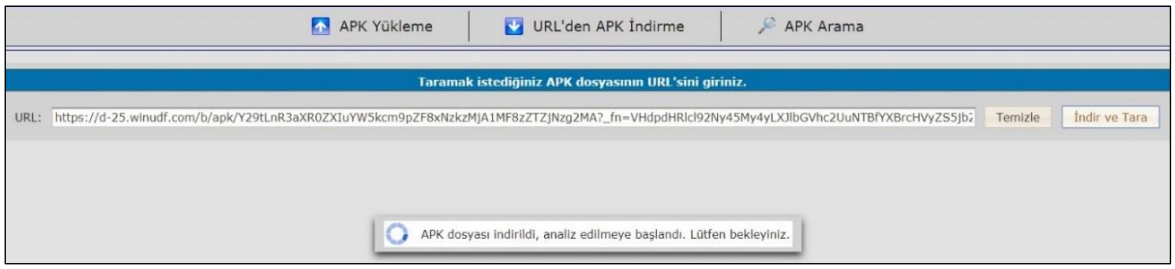


Şekil 4.16. APKPure web sitesinden APK dosyası indirme cevabı

APK yükleme arayüzünde olduğu gibi URL'den APK indirme arayüzünde de analiz edilecek APK dosyasının indirilmesi ya da analiz edilmesi sırasında kullanıcıların sayfayı değiştirmelerinin engellenmesi amacıyla tüm sayfa BlockUI gizli penceresi ile kaplanmakta ve pencere üzerinde Şekil 4.17 ve Şekil 4.18'de gösterildiği gibi durum kullanıcıya bildirilmektedir.



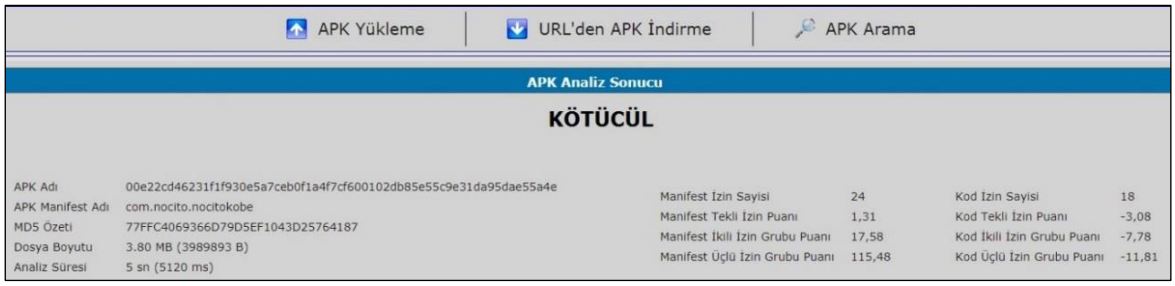
Şekil 4.17. APK dosyası indiriliyor bildirimi



Şekil 4.18. APK dosyası indirildi bildirimi

ANUAS'a yüklenen ya da sistem tarafından indirilen APK dosyaları analiz edildikten sonra Şekil 4.19 ve Şekil 4.20'de gösterildiği gibi sonuç penceresinde analiz edilen uygulamaya ilişkin bilgiler ile birlikte uygulamanın kötücül ya da iyicil olduğu kullanıcıya bildirilir.

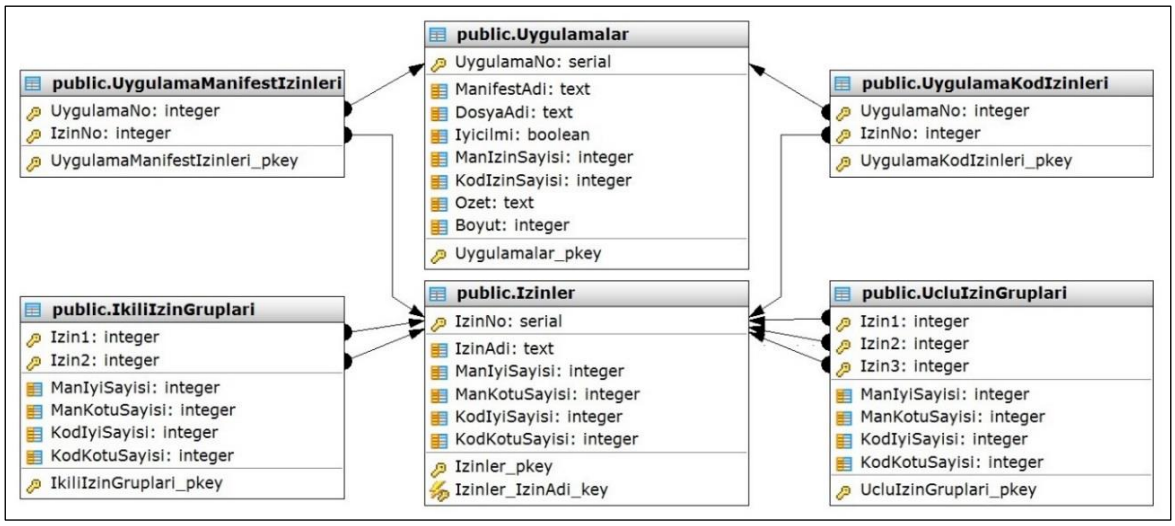
Şekil 4.19. APK analiz sonucu (İyicil)



Şekil 4.20. APK analiz sonucu (Kötücül)

4.3. Veri Tabanı Yapısı

ANUAS'ta veri tabanı olarak PostgreSQL Server yazılımının 11.5.1 sürümü kullanılmıştır. Sistemin veri tabanı yapısı Şekil 4.21'de sunulmuştur.



Şekil 4.21. Veri tabanı yapısı

Izinler tablosunda her farklı izin adı sıradaki izin numarası atanarak ayrı bir satır olarak kaydedilmektedir. İzin numarası tablonun birincil anahtarı olarak belirlenmiş olup izin adının da tekil olması zorunlu hale getirilmiştir. Tabloda ayrıca her bir iznin manifest ya da kod izni olarak kaç adet iyicil ya da kötücül uygulama tarafından kullanıldığının kaydını tutmak amacıyla ManIyiSayisi, ManKotuSayisi, KodIyiSayisi ve KodKotuSayisi sütunları bulunmaktadır. API seviyesi 28'e (Android 9 Pie) kadar olan Android işletim sistemlerinde varsayılan olarak tanımlı olan tüm izinlerin birleşimi olan 706 adet izin varsayılan olarak tabloya eklenmiştir. Bu izinler, Android Emulator yazılımında farklı Android sürümlerinin x86 işletim sistemi imajlarıyla oluşturulan sanal cihazlar ile komut satırından "adb shell pm list permissions -g" komutu girilerek elde edilmiş ve ardından birleştirilmiştir. Ayrıca eğitim aşamasında kullanılan uygulamaların AndroidManifest.xml dosyalarında tanımladıkları yeni izinler de tabloya eklenmektedir. Örnek tablo içeriği Şekil 4.22'de sunulmuştur.

IzinNo [PK] integer	IzinAdi text	ManIyiSayisi integer	ManKotuSayisi integer	KodIyiSayisi integer	KodKotuSayisi integer
223	android.permission.INTERACT_ACROSS_USERS	22	0	97	0
224	android.permission.INTERACT_ACROSS_USERS_FULL	46	0	0	0
225	android.permission.INTERNAL_DELETE_CACHE_FILES	0	0	0	0
226	android.permission.INTERNAL_SYSTEM_WINDOW	7	7	5	0
227	android.permission.INTERNET	3844	3724	3597	1280
228	android.permission.INVOKE_CARRIER_SETUP	1	0	2	0
229	android.permission.KILL_BACKGROUND_PROCESSES	125	93	16	0

Şekil 4.22. Izinler tablosu veri önizlemesi

Uygulamalar tablosunda her uygulama sıradaki uygulama numarası atanarak ayrı bir satır olarak kaydedilmektedir. Uygulama numarası tablonun birincil anahtarı olarak belirlenmiştir. Tabloda uygulama numaraları dışında uygulamaların manifest ve dosya adları, iyicil ya da kötücül olma durumları, manifest ve kod izni sayıları, SHA-256 özetleri ve bayt olarak dosya boyutları tutulmaktadır. Uygulamaların manifest adları AndroidManifest.xml dosyasında bulunan "package" bilgisinden alınmaktadır. İyicil ya da kötücül olma durumu Iyicilmi sütununda doğru ya da yanlış değeri ile tutulmaktadır. İyicil uygulamalar için bu sütuna doğru, kötücül uygulamalar için yanlış yazılmaktadır. Örnek tablo içeriği Şekil 4.23'te sunulmuştur.

UygulamaNo [PK] integer	ManifestAdi text	DosyaAdi text	Iyicilmi boolean	ManIzinSayisi integer	KodIzinSayisi integer	Ozet text	Boyut integer
3885	com.arungruang.bliuntung	Uygulamalar-İş 00110 Beli Untung_v1.7.4	true	11	19	9C8...	13897457
3886	com.facebook.work	Uygulamalar-İş 00111 Workplace by Facebook_v231.0.0.43.113	true	61	30	9C2...	43420753
3887	com.microsoft.teams	Uygulamalar-İş 00112 Microsoft Teams_v1416_1.0.0.2019072402	true	46	39	2A2...	48814094
3888	com.facebook.workchat	Uygulamalar-İş 00113 Workplace Chat by Facebook_v224.1.0.18.117	true	55	63	FB4...	46785058
3889	com.perfectlove	00002d74a9faa53f5199c910b652ef09d3a7f6bd42b693755a233635c3ffb0f4	false	11	1	000...	238808
3890	tp5x.WGt12	0000764713b286cfe7e8e76c7038c92312977712d9c5a86d504be54f3c1d025a	false	11	1	000...	139335
3891	beelon.android.alarm	00088e191503bbfbd5c56a789a71e8c718e42ea422ec73c760ee2de489e02b2e	false	4	0	000...	407515
3892	com.depositmobi	000a067df9235aea987cd1e6b7768bcc1053e640b267c5b1f0deefc18be5dbe1	false	1	0	000...	3366203

Şekil 4.23. Uygulamalar tablosu veri önizlemesi

IkiliIzinGruplari tablosunda iki elemanlı izin gruplarını oluşturan izinlerin numaraları ile birlikte her bir izin grubundaki izinlerin manifest ya da kod izni olarak birlikte kaç adet iyicil ya da kötücül uygulama tarafından kullanıldığının kaydını tutmak amacıyla ManIyiSayisi, ManKotuSayisi, KodIyiSayisi ve KodKotuSayisi sütunları bulunmaktadır. Izinler tablosunun birincil anahtarı olan izin numarası IkiliIzinGruplari tablosunun Izin1 ve Izin2 sütunlarında yabancı anahtar olarak kullanılmaktadır. Izin1 ve Izin2 sütunları birlikte IkiliIzinGruplari tablosunun birincil anahtarı olarak belirlenmiştir. Örnek tablo içeriği Şekil 4.24'te sunulmuştur.

Izin1 [PK] integer	Izin2 [PK] integer	ManIyiSayisi integer	ManKotuSayisi integer	KodIyiSayisi integer	KodKotuSayisi integer
28	227	3798	2540	3570	966
28	460	3150	1074	2922	29
28	469	2760	1945	3431	247
28	537	2732	54	808	0
227	460	3157	1497	2893	37
227	469	2788	2589	3348	278
227	537	2732	272	801	0
460	537	2692	231	767	0

Şekil 4.24. IkiliIzinGruplari tablosu veri önizlemesi

UcluIzinGruplari tablosunda üç elemanlı izin gruplarını oluşturan izinlerin numaraları ile birlikte her bir izin grubundaki izinlerin manifest ya da kod izni olarak birlikte kaç adet iyicil ya da kötücül uygulama tarafından kullanıldığının kaydını tutmak amacıyla ManIyiSayisi, ManKotuSayisi, KodIyiSayisi ve KodKotuSayisi sütunları bulunmaktadır. Izinler tablosunun birincil anahtarı olan izin numarası UcluIzinGruplari tablosunun Izin1, Izin2 ve Izin3 sütunlarında yabancı anahtar olarak kullanılmaktadır. Izin1, Izin2 ve Izin3 sütunları birlikte UcluIzinGruplari tablosunun birincil anahtarı olarak belirlenmiştir. Örnek tablo içeriği Şekil 4.25'te sunulmuştur.

Izin1 [PK] integer	Izin2 [PK] integer	Izin3 [PK] integer	ManIyiSayisi integer	ManKotuSayisi integer	KodIyiSayisi integer	KodKotuSayisi integer
28	40	227	2300	1607	2282	260
28	227	460	3150	1073	2891	28
28	227	469	2760	1944	3336	242
28	227	537	2732	54	800	0
28	460	469	2399	798	2721	16
28	460	537	2692	13	764	0
227	460	469	2406	1166	2698	18
227	460	537	2692	231	758	0

Şekil 4.25. UcluIzinGruplari tablosu veri önizlemesi

UygulamaManifestIzinleri tablosunda uygulamaların numaraları ile manifest izinlerinin numaraları eşleştirilerek uygulamaların manifest dosyalarında hangi izinleri talep ettiklerinin kaydı tutulur. Uygulamalar tablosunun birincil anahtarı olan uygulama numarası UygulamaManifestIzinleri tablosunun UygulamaNo sütununda, Izinler tablosunun birincil

anahtarı olan izin numarası ise UygulamaManifestIzinleri tablosunun IzinNo sütununda yabancı anahtar olarak kullanılmaktadır. UygulamaNo ve IzinNo sütunları birlikte UygulamaManifestIzinleri tablosunun birincil anahtarı olarak belirlenmiştir. Örnek tablo içeriği Şekil 4.26'da sunulmuştur.

	UygulamaNo [PK] integer	IzinNo [PK] integer
1	1	28
2	1	40
3	1	149
4	1	227
5	1	325
6	1	460
7	1	469
8	1	537
9	1	707
10	2	28
11	2	40
12	2	227

Şekil 4.26. UygulamaManifestIzinleri tablosu veri önizlemesi

UygulamaKodIzinleri tablosunda uygulamaların numaraları ile kod izinlerinin numaraları eşleştirilerek uygulamaların kod içerisinde hangi izinleri kullandıklarının kaydı tutulur. UygulamaManifestIzinleri tablosu ile aynı yapıya sahiptir. Örnek tablo içeriği Şekil 4.27'de sunulmuştur.

	UygulamaNo [PK] integer	IzinNo [PK] integer
1	1	10
2	1	17
3	1	28
4	1	40
5	1	110
6	1	149
7	1	195
8	1	227
9	1	270
10	1	325

Şekil 4.27. UygulamaKodIzinleri tablosu veri önizlemesi

5. TESTLER VE BULGULAR

Bu bölümde ANUAS'ın performansını değerlendirmek maksadıyla yapılan eğitim ve test faaliyetlerinde kullanılan kötücül ve iyicil uygulama veri kümeleri, yapılan testler ve bunların sonucunda elde edilen bulgular anlatılacaktır.

5.1. Kötücül Uygulama Veri Kümesi

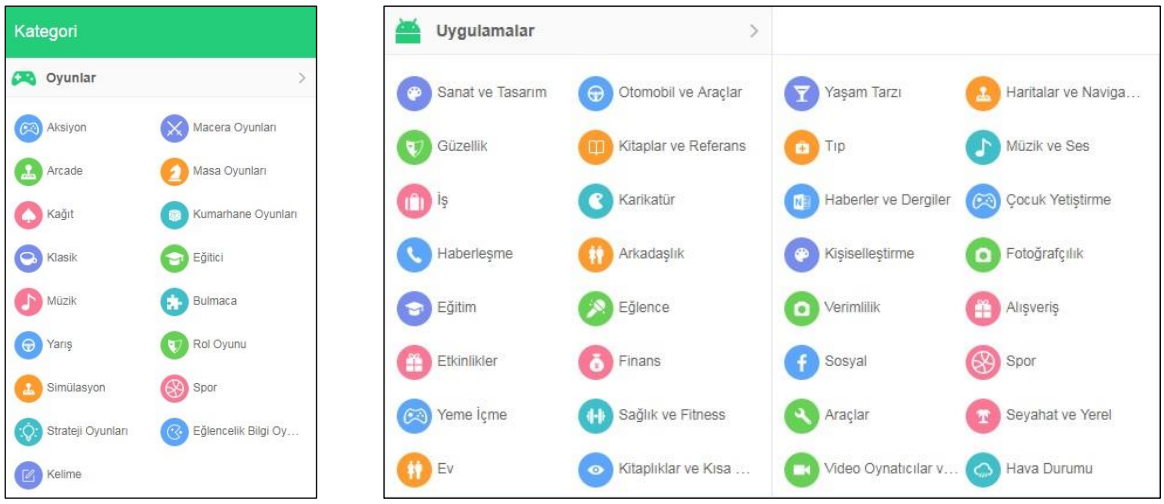
ANUAS'ın eğitim ve test aşamalarında kötücül uygulama veri kümesi olarak Drebin projesinin kötücül veri kümesi kullanılmıştır. Mobile-Sandbox projesiyle 2010 yılı Ağustos ayı ile 2012 yılı Ekim ayı arasında 123 453 adet iyicil uygulama ve toplam 179 farklı kötücül yazılım ailesinden 5560 adet kötücül uygulama toplanmış, kötücül uygulamalar akademik çalışmaları desteklemek amacıyla Drebin veri kümesi adıyla erişime açılmıştır [61, 84]. Drebin kötücül uygulama veri kümesi Türkiye'den Gazi Üniversitesi, Abant İzzet Baysal Üniversitesi, Bilkent Üniversitesi, Erzurum Teknik Üniversitesi, Hacettepe Üniversitesi, Sabancı Üniversitesi, Süleyman Demirel Üniversitesi, TÜBİTAK Siber Güvenlik Enstitüsü de dâhil olmak üzere dünya çapında pek çok üniversitede yapılan çalışmalarda kullanılmıştır [93].

Drebin veri kümesi içerisinde 5560 kötücül uygulama olmakla birlikte bunlardan 6 adedi bozuk ya da standart APK formatına uymayan dosyalardır, dolayısıyla bu uygulamaların manifest ya da kod izinleri çıkarılamamıştır. Bu nedenle bu 6 uygulama kötücül uygulama veri kümesinden çıkarılmış ve kalan 5554 uygulama ANUAS'ın eğitim ve test aşamalarında kullanılmıştır. Bu uygulamaların kötücül olma durumlarında bir değişiklik olup olmadığını kontrol etmek maksadıyla tüm uygulamalar VirusTotal API kullanılarak taratılmıştır. VirusTotal, 2004 yılında kurulmuş ve 2012 yılında Google tarafından satın alınmış, hâlihazırda 70'ten fazla anti virüs yazılımı ve URL/etki alanı kara listeleme hizmetini kullanarak kullanıcılar tarafından yüklenen dosyaları veya girilen URL adreslerini tarayan ücretsiz çevrimiçi bir web güvenlik hizmetidir. Günümüzde Fortune 500 şirketleri, hükümetler ve lider güvenlik şirketleri, akademik araştırmacılar ve pek çok son kullanıcı tarafından kullanılmakta olup 500 000'den fazla kayıtlı kullanıcısı bulunmaktadır [94, 95]. VirusTotal hizmeti literatürdeki pek çok akademik çalışmada da kullanılmıştır.

VirusTotal, kullanıcılara geliştirdikleri uygulamaların içerisine VirusTotal hizmetini bütünleştirebilmeleri için API desteği sağlamaktadır. Ücretsiz API desteği ticari olmayan ürünlerde kullanılabilmekte olup dakikada 4 adede kadar sorguya ve en fazla 32 MB'a kadar dosya yüklemeye izin verilmektedir. Ücretli API desteğiyle ise çok daha fazla sorgu yapabilme, sorgulanan dosyalara ilişkin tüm bilgileri alabilme, özeti gönderilen bir dosyayı VirusTotal'den indirebilme ve 200 MB'a kadar dosya yükleme gibi ek imkânlar sağlanmaktadır [96]. Bu tez çalışması kapsamında geçici bir süre için edinilen VirusTotal Akademik API desteğiyle günde 20 000 adede kadar sorgu yapabilme ve 200 MB'a kadar dosya yükleme imkânı elde edilmiştir. Ardından Java programlama dili kullanılarak VirusTotalde Taratma Uygulaması adlı bir uygulama geliştirilmiş, Drebin veri kümesindeki 5554 kötücül uygulamanın SHA-256 özetleri VirusTotal'e gönderilerek yeniden tarama talep edilmiş ve ardından tüm uygulamalar için rapor alınmıştır. Bu çalışmada veri kümesindeki tüm uygulamaların daha önceden taratılmış olduğu tespit edilmiş, bu nedenle hiçbir uygulama VirusTotal'e yüklenmemiştir. Alınan raporlar sonucunda veri kümesindeki tüm uygulamaların en az 3, en fazla 55, ortalama 33 adet olmak üzere VirusTotal hizmetinde bulunan anti virüs yazılımlarının bir kısmı tarafından kötücül olarak tespit edildiği görülmüştür. Taratma işleminden sonra literatürdeki başka akademik çalışmalara benzer şekilde uygulamaların %70'inin eğitim, %30'unun test için kullanılmasına karar verilmiş, bu kapsamda rastgele seçilen 3888 uygulama eğitim, 1666 uygulama test için kullanılmıştır.

5.2. İyicil Uygulama Veri Kümesi

Tez çalışması kapsamında ANUAS'ın eğitim ve test aşamalarında kullanılacak iyicil uygulama veri kümesi olarak Drebin kötücül veri kümesi ile eşit sayıda iyicil uygulama kullanılmasına ve kötücül veri kümesi ile benzer şekilde iyicil uygulamaların toplamda 3888 adedinin eğitim, 1666 adedinin test için ayrılmasına karar verilmiştir. İyicil uygulama veri kümesini oluşturmak maksadıyla Java programlama dili kullanılarak APKPure Apk İndirme Uygulaması geliştirilmiştir. Bu uygulama ile 2019 yılı Temmuz ayı içerisinde APKPure uygulama mağazasında bulunan 49 adet uygulama kategorisinin her birinden yaklaşık olarak eşit sayıda uygulama toplanmıştır. APKPure uygulama mağazasının oyun kategorileri Şekil 5.1a'da, uygulama kategorileri Şekil 5.1b'de gösterilmiştir.



a b

Şekil 5.1. a) Oyun kategorileri b) Uygulama kategorileri

APKPure Apk İndirme Uygulaması ile öncelikle her uygulama kategorisinde en çok indirilen uygulamaların indirme bağlantıları toplanmış ve ardından Uygulamalar-Finans kategorisi haricindeki her bir kategoriden en fazla 150 MB boyutlu 130'ar adet uygulama indirilmiştir. APKPure uygulama mağazasında Uygulamalar-Finans kategorisinde en fazla 150 MB boyutlu sadece 105 uygulama bulunması nedeniyle sadece bu uygulamalar indirilebilmiştir. Ardından indirilen tüm uygulamalar geliştirilen VirusTotalde Taratma Uygulaması kullanılarak taratılmış, VirusTotal hizmetinde bir tane bile anti virüs yazılımı tarafından kötücül olarak tespit edilen uygulamalar kötücül kabul edilerek iyicil uygulama veri kümesinden çıkarılmıştır. Taratma sonucunda eksik kalan kategoriler için sıradaki dosyalar indirilmiş ve taratılmıştır. Bu işlemler sonucunda toplamda 6362 adet APK ya da XAPK dosyası indirilmiş ve taratılmış, bunlardan 2295 adedi VirusTotal veri tabanında bulunmaması nedeniyle VirusTotal'e yüklenmiştir. Tüm dosyalardan 526 adedinin aslında kötücül olduğu tespit edilmiş, geriye kalan 5836 dosyadan her kategoride en çok indirilen 5554 adedi iyicil veri kümesi olarak kullanılmıştır. İyicil veri kümesinin oluşumu ayrıntılı olarak Çizelge 5.1'de gösterilmiştir.

Çizelge 5.1. İyicil veri kümesinin oluşumu

Kategori	İndirilen Apk Sayısı	Kötücül Apk Sayısı	İyicil Apk Sayısı	Kullanılan Toplam Apk Sayısı	Eğitim için Kullanılan Apk Sayısı	Test için Kullanılan Apk Sayısı
Oyunlar-Aksiyon	130	15	115	113	79	34
Oyunlar-Arcade	130	15	115	114	80	34
Oyunlar-Bulmaca	130	10	120	114	80	34

Çizelge 5.1. (devam) İyicil veri kümesinin oluşumu

Oyunlar-Eğitici	130	11	119	113	79	34
Oyunlar-Eğlencelik Bilgi Oyunları	130	7	123	114	80	34
Oyunlar-Kâğıt	130	14	116	113	79	34
Oyunlar-Kelime	130	10	120	114	80	34
Oyunlar-Klasik	130	10	120	114	79	35
Oyunlar-Kumarhane Oyunları	131	17	114	114	80	34
Oyunlar-Macera Oyunları	140	26	114	114	80	34
Oyunlar-Masa Oyunları	130	13	117	114	79	35
Oyunlar-Müzik	130	16	114	114	80	34
Oyunlar-Rol Oyunu	130	13	117	113	79	34
Oyunlar-Simülasyon	130	15	115	114	80	34
Oyunlar-Spor	130	14	116	114	80	34
Oyunlar-Strateji Oyunları	130	11	119	114	80	34
Oyunlar-Yarış	130	11	119	114	80	34
Uygulamalar-Alışveriş	130	7	123	114	79	35
Uygulamalar-Araçlar	130	11	119	113	79	34
Uygulamalar-Arkadaşlık	130	10	120	114	80	34
Uygulamalar-Çocuk Yetiştirme	130	9	121	114	79	35
Uygulamalar-Eğitim	130	7	123	114	80	34
Uygulamalar-Eğlence	133	19	114	114	80	34
Uygulamalar-Etkinlikler	130	5	125	114	80	34
Uygulamalar-Ev	130	8	122	114	80	34
Uygulamalar-Finans	105	10	95	94	67	27
Uygulamalar-Fotoğrafçılık	133	19	114	114	80	34
Uygulamalar-Güzellik	130	10	120	113	79	34
Uygulamalar-Haberler ve Dergiler	130	11	119	114	79	35
Uygulamalar-Haberleşme	130	9	121	114	80	34
Uygulamalar-Haritalar ve Navigasyon	130	8	122	114	80	34
Uygulamalar-Hava Durumu	130	15	115	113	79	34
Uygulamalar-İş	130	2	128	113	79	34
Uygulamalar-Karikatür	130	8	122	114	80	34
Uygulamalar-Kişiselleştirme	130	12	118	114	80	34
Uygulamalar-Kitaplar ve Referans	130	10	120	114	80	34
Uygulamalar-Kitaplıklar ve Kısa Sunum	130	7	123	114	80	34
Uygulamalar-Müzik ve Ses	130	8	122	113	79	34
Uygulamalar-Otomobil ve Araçlar	130	6	124	113	79	34
Uygulamalar-Sağlık ve Fitness	130	9	121	114	80	34
Uygulamalar-Sanat ve Tasarım	130	13	117	114	79	35
Uygulamalar-Seyahat ve Yerel	130	10	120	114	80	34
Uygulamalar-Sosyal	130	10	120	114	79	35
Uygulamalar-Spor	130	9	121	113	79	34
Uygulamalar-Tıp	130	6	124	114	80	34
Uygulamalar-Verimlilik	130	8	122	114	80	34
Uygulamalar-Video Oynatıcılar ve Düzenleyiciler	130	9	121	114	80	34
Uygulamalar-Yaşam Tarzı	130	9	121	114	80	34
Uygulamalar-Yeme İçme	130	4	126	113	79	34
TOPLAM	6362	526	5836	5554	3888	1666

5.3. Eğitim Aşaması

Eğitim aşamasında iyicil uygulama veri kümesindeki her kategoriden rastgele seçilen 79-80 uygulamadan oluşan toplam 3888 iyicil uygulama ve Drebin kötücül uygulama veri kümesinden rastgele seçilen 3888 kötücül uygulama kullanılmıştır. Bu dosyalar çözülerek manifest ve kod izinleri çıkarılmış, ardından veri tabanına eklenmiştir. Eğitim aşamasından önce varsayılan olarak 706 adet izin adı veri tabanına eklenmiştir. Eğitim aşamasında 3888 iyicil ve 3888 kötücül uygulama veri tabanına eklendikten sonra bu uygulamaların AndroidManifest.xml dosyalarında toplamda 2906 adet yeni izin tanımladıkları ve bu izinlerin veri tabanına eklendiği, Izinler tablosunda toplamda 3612 adet izin bulunduğu görülmüştür. Yeni tanımlanan 2906 adet izinden 2794 adedinin sadece iyicil uygulamalarda, 86 adedinin sadece kötücül uygulamalarda, 26 adedinin ise her iki türdeki uygulamalarda ortak olarak bulunduğu tespit edilmiştir. Bu durum yeni tanımlanan izinlerin çok büyük çoğunlukla iyicil uygulamalar tarafından tanımlandığını göstermekte olup bunun iyicil uygulamalar ile kötücül uygulamaların toplanma zamanlarının oldukça farklı olmasından kaynaklanabileceği değerlendirilmektedir. Eğitim aşamasında aynı zamanda Izinler tablosuna varsayılan olarak eklenen 706 adet izinden 251 adedinin 7776 uygulamanın hiçbiri tarafından manifest izni olarak talep edilmediği ya da kod içerisinde kullanılmadığı tespit edilmiştir. Eğitim aşamasında iyicil ve kötücül uygulamalar tarafından talep edilen ve kullanılan izinlere ilişkin istatistikler Çizelge 5.2'de gösterilmiştir.

Çizelge 5.2. Talep edilen ve kullanılan izin istatistikleri

	İyicil Uygulamalar	Kötücül Uygulamalar
En fazla izin talep eden uygulamanın talep ettiği izin sayısı	212	38
Hiç izin talep etmeyen uygulama sayısı	21	17
Ortalama talep edilen izin sayısı	14,71	11,79
En fazla izin kullanan uygulamanın kullandığı izin sayısı	285	21
Hiç izin kullanmayan uygulama sayısı	70	1462
Ortalama kullanılan izin sayısı	21,67	3,10

Bu istatistiklere göre en fazla izin talep eden uygulamaların talep ettikleri izin sayıları arasında büyük bir fark olsa da ortalamada iyicil uygulamalar ve kötücül uygulamalar birbirine yakın sayıda izin talep etmektedir. Ancak hiç izin kullanmayan uygulama sayısı ve ortalama kullanılan izin sayısı bakımından iyicil uygulamalar ve kötücül uygulamalar

arasında büyük bir fark olduğu görülmektedir. Bu sonuca göre kötücül uygulamaların kod içerisinde iyicil uygulamalara göre çok daha az kod ismi bulundurduğu ve bunun Drebin veri kümesindeki uygulamaların toplanma zamanına ya da kötücül uygulamaların kodlarını ve amaçlarını gizlemeye çalışmalarına bağlı olabileceği değerlendirilmektedir.

ANUAS'ta kullanılan sorgulama yöntemine göre veri tabanındaki Izinler tablosunun ManIyiSayisi, ManKotuSayisi, KodIyiSayisi ve KodKotuSayisi sütunlarından her biri için değeri sıfırdan büyük olan satırları kapsayan birer eşleşme tablosu oluşturulmaktadır. Eşleşme tablolarının büyüklüklerini göstermek amacıyla Izinler tablosunun ManIyiSayisi, ManKotuSayisi, KodIyiSayisi ve KodKotuSayisi sütunlarındaki değerlerin dağılımı Çizelge 5.3'te sunulmuştur.

Çizelge 5.3. Izinler tablosundaki izinlerin talep edilme/kullanılma durumu

Toplam İzin Sayısı: 3612	ManIyiSayisi	ManKotuSayisi	KodIyiSayisi	KodKotuSayisi
Değeri 0 olan satır sayısı	437	3382	3148	3536
Değeri 0'dan büyük olan satır sayısı	3175	230	464	76

Çizelgeye göre iyicil uygulamalar AndroidManifest.xml dosyalarında 3175 adet farklı izin, kötücül uygulamalar ise 230 adet farklı izin talep etmişlerdir. Kod izinlerine bakıldığında ise iyicil uygulamaların kod içerisinde 464 adet farklı izin, kötücül uygulamaların 76 adet farklı izin kullandıkları görülmektedir. Bu durumda tekilIzinManIyiSayilari eşleşme tablosu 3175, tekilIzinManKotuSayilari eşleşme tablosu 230, tekilIzinKodIyiSayilari eşleşme tablosu 464 ve tekilIzinKodKotuSayilari eşleşme tablosu 76 adet izin numarası-değer eşleşmesi içermektedir.

Eğitim aşamasında kullanılan iyicil ve kötücül uygulamalar tarafından en fazla talep edilen onar izin Çizelge 5.4'te sunulmuştur. Çizelgede adet ilgili izni talep eden iyicil ya da kötücül uygulama sayısını, yüzdelik oran ise ilgili izni talep eden iyicil/kötücül uygulama sayısının toplam iyicil/kötücül uygulama sayısına göre yüzdesini ifade etmektedir. Bu çizelgede ve sonraki çizelgelerde izin adlarını yazarken çizelgelerin daha okunaklı olmasını sağlamak amacıyla android.permission ile başlayan izin adlarındaki android.permission ifadesi silinmiştir. Örnek olarak android.permission.INTERNET izni çizelgelerde sadece INTERNET şeklinde yazılmıştır.

Çizelge 5.4. En fazla talep edilen onar izin

İYİCİL UYGULAMALAR			KÖTÜCÜL UYGULAMALAR		
İzin Adı	Frekans		İzin Adı	Frekans	
	Adet	Yüzdelik Oran		Adet	Yüzdelik Oran
INTERNET	3844	%98,87	INTERNET	3724	%95,78
ACCESS_NETWORK_STATE	3799	%97,71	READ_PHONE_STATE	3459	%88,97
WAKE_LOCK	3159	%81,25	WRITE_EXTERNAL_STORAGE	2611	%67,16
WRITE_EXTERNAL_STORAGE	2796	%71,91	ACCESS_NETWORK_STATE	2547	%65,51
com.google.android.c2dm.permission.RECEIVE	2732	%70,27	SEND_SMS	2116	%54,42
ACCESS_WIFI_STATE	2301	%59,18	RECEIVE_BOOT_COMPLETED	1853	%47,66
com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE	2136	%54,94	ACCESS_WIFI_STATE	1695	%43,60
VIBRATE	1959	%50,39	RECEIVE_SMS	1513	%38,91
READ_EXTERNAL_STORAGE	1947	%50,08	WAKE_LOCK	1513	%38,91
com.android.vending.BILLING	1919	%49,36	READ_SMS	1488	%38,27

Çizelgeye göre android.permission.INTERNET izni hem iyicil hem de kötücül uygulamalar tarafından en fazla talep edilen izindir. Ayrıca ACCESS_NETWORK_STATE, WAKE_ LOCK, WRITE_ EXTERNAL_STORAGE, ACCESS_WIFI_STATE izinleri de talep edilme oranları farklı olsa da her iki türdeki uygulamalar tarafından en fazla talep edilen onar izin arasındadır.

Bunların dışında com.google.android.c2dm.permission.RECEIVE, com.google.android. finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE, VIBRATE, READ_ EXTERNAL_STORAGE ve com.android.vending.BILLING izinleri sadece iyicil uygulamalarda en fazla talep edilen on izin arasındadır. READ_PHONE_STATE, SEND_SMS, RECEIVE_BOOT_COMPLETED, RECEIVE_SMS ve READ_SMS izinleri ise sadece kötücül uygulamalarda en fazla talep edilen on izin arasında bulunmaktadır.

Eğitim aşamasında kullanılan iyicil ve kötücül uygulamalar tarafından kod içerisinde en fazla kullanılan onar izin Çizelge 5.5'te sunulmuştur.

Çizelge 5.5. Kod içerisinde en fazla kullanılan onar izin

İYİCİL UYGULAMALAR			KÖTÜCÜL UYGULAMALAR		
İzin Adı	Frekans		İzin Adı	Frekans	
	Adet	Yüzdelik Oran		Adet	Yüzdelik Oran
ACCESS_NETWORK_STATE	3710	%95,42	INTERNET	1280	%32,92
ACCESS_FINE_LOCATION	3688	%94,86	ACCESS_NETWORK_STATE	1091	%28,06
ACCESS_COARSE_LOCATION	3673	%94,47	ACCESS_COARSE_LOCATION	1056	%27,16
INTERNET	3597	%92,52	READ_PHONE_STATE	1044	%26,85
WRITE_EXTERNAL_STORAGE	3478	%89,45	ACCESS_FINE_LOCATION	1043	%26,83
UPDATE_DEVICE_STATS	3159	%81,25	com.android.launcher.action.INSTALL_SHORTCUT	804	%20,68
com.google.android.providers.gsf.permission.READ_GSERVICES	3131	%80,53	android.provider.Telephony.SMS_RECEIVED	570	%14,66
WAKE_LOCK	2933	%75,44	com.google.android.maps	527	%13,55
com.google.android.c2dm.permission.SEND	2622	%67,44	VIBRATE	448	%11,52
CAMERA	2616	%67,28	ACCESS_WIFI_STATE	397	%10,21

Çizelgeye göre iyicil uygulamalar ile kötücül uygulamalar tarafından kod içerisinde en fazla kullanılan onar iznin kullanım oranları arasında büyük bir fark bulunmaktadır. Çizelge kötücül uygulamalarda kod içerisinde kullanılan izinler olarak belirli izinlerde yoğunlaşma olmadığını göstermektedir. Buna karşın kullanım oranları arasında büyük bir fark olsa da ACCESS_NETWORK_STATE, ACCESS_FINE_LOCATION, ACCESS_COARSE_ LOCATION, INTERNET izinleri her iki türdeki uygulamalar tarafından kod içerisinde en fazla kullanılan onar izin arasındadır.

Bunların dışında WRITE_EXTERNAL_STORAGE, UPDATE_DEVICE_STATS, com. google.android.providers.gsf.permission.READ_GSERVICES, WAKE_LOCK, com. google.android.c2dm.permission.SEND ve CAMERA izinleri sadece iyicil uygulamalarda en fazla kullanılan on izin arasındadır. READ_PHONE_STATE, com.android.launcher. action.INSTALL_SHORTCUT, android.provider.Telephony.SMS_RECEIVED, com. google.android.maps, VIBRATE ve ACCESS_WIFI_STATE izinleri ise sadece kötücül uygulamalarda en fazla kullanılan on izin arasında bulunmaktadır.

Eğitim aşamasında 7776 uygulamanın veri tabanına eklenmesinden sonra IkiliIzinGruplari tablosunda toplamda 183 353 adet izin grubu oluşmuştur. İkili izin gruplarına ilişkin eşleşme tablolarının büyüklüklerini göstermek amacıyla IkiliIzinGruplari tablosunun ManIyiSayisi,

ManKotuSayisi, KodIyiSayisi ve KodKotuSayisi sütunlarındaki değerlerin dağılımı Çizelge 5.6'da sunulmuştur.

Çizelge 5.6. İkili izin gruplarının talep edilme/kullanılma durumu

Toplam Grup Sayısı: 183353	ManIyiSayisi	ManKotuSayisi	KodIyiSayisi	KodKotuSayisi
Değeri 0 olan satır sayısı	38494	177170	130439	182506
Değeri 0'dan büyük olan satır sayısı	144 859	6183	52 914	847

Çizelgeye göre eğitimden sonra ikiliIzinGrubuManIyiSayilari eşleşme tablosu 144 859, ikiliIzinGrubuManKotuSayilari eşleşme tablosu 6183, ikiliIzinGrubuKodIyiSayilari eşleşme tablosu 52 914 ve ikiliIzinGrubuKodKotuSayilari eşleşme tablosu 847 adet izin numaraları-değer eşleşmesi içermektedir.

Eğitimde kullanılan iyicil ve kötücül uygulamalar tarafından en fazla talep edilen onar adet ikili izin grubu Çizelge 5.7'de sunulmuştur.

Çizelge 5.7. En fazla talep edilen onar adet ikili izin grubu

İYİCİL UYGULAMALAR			KÖTÜCÜL UYGULAMALAR		
İzin Grubu Elemanları	Frekans		İzin Grubu Elemanları	Frekans	
	Adet	Yüzdelik Oran		Adet	Yüzdelik Oran
ACCESS_NETWORK_STATE INTERNET	3798	%97,69	INTERNET READ_PHONE_STATE	3436	%88,37
INTERNET WAKE_LOCK	3157	%81,20	INTERNET WRITE_EXTERNAL_STORAGE	2589	%66,59
ACCESS_NETWORK_STATE WAKE_LOCK	3150	%81,02	ACCESS_NETWORK_STATE INTERNET	2540	%65,33
INTERNET WRITE_EXTERNAL_STORAGE	2788	%71,71	READ_PHONE_STATE WRITE_EXTERNAL_STORAGE	2535	%65,20
ACCESS_NETWORK_STATE WRITE_EXTERNAL_STORAGE	2760	%70,99	ACCESS_NETWORK_STATE READ_PHONE_STATE	2456	%63,17
ACCESS_NETWORK_STATE com.google.android.c2dm.permission.RECEIVE	2732	%70,27	INTERNET SEND_SMS	2005	%51,57
INTERNET com.google.android.c2dm.permission.RECEIVE	2732	%70,27	ACCESS_NETWORK_STATE WRITE_EXTERNAL_STORAGE	1945	%50,03

Çizelge 5.7. (devam) En fazla talep edilen onar adet ikili izin grubu

WAKE_LOCK com.google.android.c2dm.permission.RECEIVE	2692	%69,24	READ_PHONE_STATE SEND_SMS	1887	%48,53
WAKE_LOCK WRITE_EXTERNAL_STORAGE	2408	%61,93	INTERNET RECEIVE_BOOT_COMPLETED	1847	%47,51
ACCESS_NETWORK_STATE ACCESS_WIFI_STATE	2300	%59,16	READ_PHONE_STATE RECEIVE_BOOT_COMPLETED	1817	%46,73

Çizelgeye göre ACCESS_NETWORK_STATE – INTERNET, INTERNET – WRITE_ EXTERNAL_STORAGE ve ACCESS_NETWORK_STATE – WRITE_EXTERNAL_ STORAGE izin grupları her iki türdeki uygulamalar tarafından en fazla talep edilen onar ikili izin grubu arasındadır.

Eğitimde kullanılan iyicil ve kötücül uygulamalar tarafından kod içerisinde en fazla kullanılan onar adet ikili izin grubu Çizelge 5.8'de sunulmuştur.

Çizelge 5.8. Kod içerisinde en fazla kullanılan onar adet ikili izin grubu

İYİCİL UYGULAMALAR			KÖTÜCÜL UYGULAMALAR		
İzin Grubu Elemanları	Frekans		İzin Grubu Elemanları	Frekans	
	Adet	Yüzdelik Oran		Adet	Yüzdelik Oran
ACCESS_COARSE_LOCATION ACCESS_FINE_LOCATION	3671	%94,42	ACCESS_COARSE_LOCATION ACCESS_FINE_LOCATION	983	%25,28
ACCESS_FINE_LOCATION ACCESS_NETWORK_STATE	3630	%93,36	ACCESS_NETWORK_STATE INTERNET	966	%24,85
ACCESS_COARSE_LOCATION ACCESS_NETWORK_STATE	3618	%93,06	INTERNET READ_PHONE_STATE	908	%23,35
ACCESS_NETWORK_STATE INTERNET	3570	%91,82	ACCESS_NETWORK_STATE READ_PHONE_STATE	866	%22,27
ACCESS_FINE_LOCATION INTERNET	3520	%90,53	ACCESS_COARSE_LOCATION INTERNET	773	%19,88
ACCESS_COARSE_LOCATION INTERNET	3513	%90,35	ACCESS_FINE_LOCATION INTERNET	703	%18,08
ACCESS_NETWORK_STATE WRITE_EXTERNAL_STORAGE	3431	%88,25	ACCESS_COARSE_LOCATION ACCESS_NETWORK_STATE	700	%18,00
ACCESS_FINE_LOCATION WRITE_EXTERNAL_STORAGE	3418	%87,91	ACCESS_FINE_LOCATION ACCESS_NETWORK_STATE	678	%17,44

Çizelge 5.8. (devam) Kod içerisinde en fazla kullanılan onar adet ikili izin grubu

ACCESS_COARSE_LOCATION WRITE_EXTERNAL_STORAGE	3408	%87,65		ACCESS_COARSE_LOCATION READ_PHONE_STATE	658	%16,92
INTERNET WRITE_EXTERNAL_STORAGE	3348	%86,11		ACCESS_FINE_LOCATION READ_PHONE_STATE	637	%16,38

Çizelgeye göre iyicil uygulamalar ile kötücül uygulamalar tarafından kod içerisinde en fazla kullanılan onar ikili iznin kullanım oranları arasında da büyük bir fark bulunmaktadır. Buna karşın kullanım oranları arasında büyük bir fark olsa da ACCESS_COARSE_LOCATION – ACCESS_FINE_LOCATION, ACCESS_FINE_LOCATION – ACCESS_NETWORK_ STATE, ACCESS_COARSE_LOCATION – ACCESS_NETWORK_STATE, ACCESS_ NETWORK_STATE – INTERNET, ACCESS_FINE_LOCATION – INTERNET ve ACCESS_COARSE_LOCATION – INTERNET izin grupları her iki türdeki uygulamalar tarafından kod içerisinde en fazla kullanılan onar ikili izin grubu arasındadır.

Eğitim aşamasında 7776 uygulamanın veri tabanına eklenmesinden sonra UcluIzinGruplari tablosunda toplamda 9 899 256 adet izin grubu oluşmuştur. Üçlü izin gruplarına ilişkin eşleşme tablolarının büyüklüklerini göstermek amacıyla UcluIzinGruplari tablosunun ManIyiSayisi, ManKotuSayisi, KodIyiSayisi ve KodKotuSayisi sütunlarındaki değerlerin dağılımı Çizelge 5.9'da sunulmuştur.

Çizelge 5.9. Üçlü izin gruplarının talep edilme/kullanılma durumu

Toplam Grup Sayısı: 9 899 256	ManIyiSayisi	ManKotuSayisi	KodIyiSayisi	KodKotuSayisi
Değeri 0 olan satır sayısı	3 892 123	9 812 521	5 576 792	9 893 788
Değeri 0'dan büyük olan satır sayısı	6 007 133	86 735	4 322 464	5468

Çizelgeye göre eğitimden sonra ucluIzinGrubuManIyiSayilari eşleşme tablosu 6 007 133, ucluIzinGrubuManKotuSayilari eşleşme tablosu 86 735, ucluIzinGrubuKodIyiSayilari eşleşme tablosu 4 322 464 ve ucluIzinGrubuKodKotuSayilari eşleşme tablosu 5468 adet izin numaraları-değer eşleşmesi içermektedir.

Eğitimde kullanılan iyicil ve kötücül uygulamalar tarafından en fazla talep edilen onar adet üçlü izin grubu Çizelge 5.10'da sunulmuştur.

Çizelge 5.10. En fazla talep edilen onar adet üçlü izin grubu

İYİCİL UYGULAMALAR			KÖTÜCÜL UYGULAMALAR		
İzin Grubu Elemanları	Frekans		İzin Grubu Elemanları	Frekans	
	Adet	Yüzdelik Oran		Adet	Yüzdelik Oran
ACCESS_NETWORK_STATE INTERNET WAKE_LOCK	3150	%81,02	INTERNET READ_PHONE_STATE WRITE_EXTERNAL_STORAGE	2526	%64,97
ACCESS_NETWORK_STATE INTERNET WRITE_EXTERNAL_STORAGE	2760	%70,99	ACCESS_NETWORK_STATE INTERNET READ_PHONE_STATE	2449	%62,99
ACCESS_NETWORK_STATE INTERNET com.google.android.c2dm.permission.RECEIVE	2732	%70,27	ACCESS_NETWORK_STATE INTERNET WRITE_EXTERNAL_STORAGE	1944	%50,00
ACCESS_NETWORK_STATE WAKE_LOCK com.google.android.c2dm.permission.RECEIVE	2692	%69,24	ACCESS_NETWORK_STATE READ_PHONE_STATE WRITE_EXTERNAL_STORAGE	1917	%49,31
INTERNET WAKE_LOCK com.google.android.c2dm.permission.RECEIVE	2692	%69,24	INTERNET READ_PHONE_STATE SEND_SMS	1871	%48,12
INTERNET WAKE_LOCK WRITE_EXTERNAL_STORAGE	2406	%61,88	INTERNET READ_PHONE_STATE RECEIVE_BOOT_COMPLETED	1812	%46,60
ACCESS_NETWORK_STATE WAKE_LOCK WRITE_EXTERNAL_STORAGE	2399	%61,70	ACCESS_WIFI_STATE INTERNET READ_PHONE_STATE	1636	%42,08
ACCESS_NETWORK_STATE ACCESS_WIFI_STATE INTERNET	2300	%59,16	ACCESS_NETWORK_STATE INTERNET RECEIVE_BOOT_COMPLETED	1630	%41,92
ACCESS_NETWORK_STATE INTERNET com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE	2136	%54,94	ACCESS_NETWORK_STATE READ_PHONE_STATE RECEIVE_BOOT_COMPLETED	1619	%41,64
ACCESS_NETWORK_STATE WAKE_LOCK com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE	2109	%54,24	ACCESS_NETWORK_STATE ACCESS_WIFI_STATE INTERNET	1607	%41,33

Çizelgeye göre ACCESS_NETWORK_STATE – INTERNET – WRITE_EXTERNAL_STORAGE ve ACCESS_NETWORK_STATE – ACCESS_WIFI_STATE – INTERNET

izin grupları her iki türdeki uygulamalar tarafından en fazla talep edilen onar üçlü izin grubu arasındadır.

Eğitimde kullanılan iyicil ve kötücül uygulamalar tarafından kod içerisinde en fazla kullanılan onar adet üçlü izin grubu Çizelge 5.11'de sunulmuştur.

Çizelge 5.11. Kod içerisinde en fazla kullanılan onar adet üçlü izin grubu

İYİCİL UYGULAMALAR			KÖTÜCÜL UYGULAMALAR		
İzin Grubu Elemanları	Frekans		İzin Grubu Elemanları	Frekans	
	Adet	Yüzdelik Oran		Adet	Yüzdelik Oran
ACCESS_COARSE_LOCATION ACCESS_FINE_LOCATION ACCESS_NETWORK_STATE	3617	%93,03	ACCESS_NETWORK_STATE INTERNET READ_PHONE_STATE	760	%19,55
ACCESS_COARSE_LOCATION ACCESS_FINE_LOCATION INTERNET	3512	%90,33	ACCESS_COARSE_LOCATION ACCESS_FINE_LOCATION INTERNET	700	%18,00
ACCESS_FINE_LOCATION ACCESS_NETWORK_STATE INTERNET	3503	%90,10	ACCESS_COARSE_LOCATION ACCESS_NETWORK_STATE INTERNET	632	%16,26
ACCESS_COARSE_LOCATION ACCESS_NETWORK_STATE INTERNET	3496	%89,92	ACCESS_COARSE_LOCATION ACCESS_FINE_LOCATION ACCESS_NETWORK_STATE	627	%16,13
ACCESS_COARSE_LOCATION ACCESS_FINE_LOCATION WRITE_EXTERNAL_STORAGE	3406	%87,60	ACCESS_COARSE_LOCATION ACCESS_NETWORK_STATE READ_PHONE_STATE	614	%15,79
ACCESS_FINE_LOCATION ACCESS_NETWORK_STATE WRITE_EXTERNAL_STORAGE	3388	%87,14	ACCESS_FINE_LOCATION ACCESS_NETWORK_STATE READ_PHONE_STATE	591	%15,20
ACCESS_COARSE_LOCATION ACCESS_NETWORK_STATE WRITE_EXTERNAL_STORAGE	3380	%86,93	ACCESS_COARSE_LOCATION ACCESS_FINE_LOCATION READ_PHONE_STATE	585	%15,05
ACCESS_NETWORK_STATE INTERNET WRITE_EXTERNAL_STORAGE	3336	%85,80	ACCESS_COARSE_LOCATION INTERNET READ_PHONE_STATE	581	%14,94
ACCESS_FINE_LOCATION INTERNET WRITE_EXTERNAL_STORAGE	3309	%85,11	ACCESS_FINE_LOCATION ACCESS_NETWORK_STATE INTERNET	561	%14,43
ACCESS_COARSE_LOCATION INTERNET WRITE_EXTERNAL_STORAGE	3306	%85,03	ACCESS_FINE_LOCATION INTERNET READ_PHONE_STATE	509	%13,09

Çizelgeye göre iyicil uygulamalar ile kötücül uygulamalar tarafından kod içerisinde en fazla kullanılan onar üçlü iznin kullanım oranları arasında da büyük bir fark bulunmaktadır. Buna karşın kullanım oranları arasında büyük bir fark olsa da ACCESS_COARSE_LOCATION – ACCESS_FINE_LOCATION – ACCESS_NETWORK_STATE, ACCESS_COARSE_ LOCATION – ACCESS_FINE_LOCATION – INTERNET, ACCESS_FINE_LOCATION – ACCESS_NETWORK_STATE – INTERNET, ACCESS_COARSE_LOCATION – ACCESS_NETWORK_STATE – INTERNET izin grupları her iki türdeki uygulamalar tarafından kod içerisinde en fazla kullanılan onar üçlü izin grubu arasındadır.

5.4. Analiz Aşaması

Analiz aşamasında iyicil uygulama veri kümesindeki her kategoriden rastgele seçilen 34-35 uygulamadan oluşan toplam 1666 iyicil uygulama ve Drebin kötücül uygulama veri kümesinden rastgele seçilen 1666 kötücül uygulama kullanılmıştır. Tüm uygulamalar tek tek çözülerek manifest ve kod izinleri çıkarılmış, bu izinlerden veri tabanında olanların izin numaraları belirlenmiş, ardından uygulamayı puanlandırma işlemine başlanmıştır. Her bir uygulama için tekil manifest izinleri puanı, tekil kod izinleri puanı, ikili manifest izin grupları puanı, ikili kod izin grupları puanı, üçlü manifest izin grupları puanı ve üçlü kod izin grupları puanı hesaplanmıştır.

ANUAS'ın puanlara göre analiz performansını değerlendirmek amacıyla karmaşıklık matrisi kullanılmıştır. Karmaşıklık matrisi bir sistem ya da algoritma tarafından yapılan tahminlerin ya da sınıflandırmaların başarısı hakkında bilgi sağlayan bir ölçüm aracıdır. Matriste gerçek durum ile sistemin verdiği sonuç karşılaştırılmaktadır. ANUAS'ta analiz edilen uygulamanın kötücül olup olmadığı sorusuna göre kullanılan karmaşıklık matrisi Çizelge 5.12'de sunulmuştur.

Çizelge 5.12. Karmaşıklık matrisi

		ANUAS ANALİZ SONUCU	
		KÖTÜCÜL	İYİCİL
GERÇEK DURUM	KÖTÜCÜL	Doğru-Pozitif (DP)	Yanlış-Negatif (YN)
	İYİCİL	Yanlış-Pozitif (YP)	Doğru-Negatif (DN)

Çizelgeye göre gerçekte kötücül olan bir uygulamanın ANUAS tarafından da kötücül olarak tespit edilmesi doğru-pozitif bir durumdur. Uygulamanın kötücül olup olmadığı sorusunun cevabı pozitiftir ve ANUAS bunu doğru şekilde tespit etmiştir. Gerçekte kötücül olan bir uygulamanın ANUAS tarafından iyicil olarak tespit edilmesi yanlış-negatif, gerçekte iyicil olan bir uygulamanın ANUAS tarafından kötücül olarak tespit edilmesi yanlış-pozitif ve gerçekte iyicil olan bir uygulamanın ANUAS tarafından da iyicil olarak tespit edilmesi ise doğru-negatif bir durumdur.

Sistemlerin veya algoritmaların performansı karmaşıklık matrisi ile doğruluk, hassasiyet, özgüllük gibi özelliklere göre değerlendirilebilir. Doğruluk bir sistemin yaptığı doğru sınıflandırmanın toplama bölümüdür (Eş. 5.1). Bu tez çalışmasında doğruluk ANUAS'ın çıkardığı analiz sonuçlarının doğru olma oranını gösterir. Hassasiyet doğru olarak belirlenen pozitif durumların tüm pozitif durumlara bölümüdür (Eş. 5.2). ANUAS'ın gerçekte kötücül olan uygulamaları kötücül olarak tespit edebilme başarısını gösterir. Özgüllük doğru olarak belirlenen negatif durumların tüm negatif durumlara bölümüdür (Eş. 5.3) ANUAS'ın gerçekte iyicil olan uygulamaları iyicil olarak tespit edebilme başarısını gösterir.

$$\text{Doğruluk} = (DP+DN) / (DP+YN+YP+DN) \quad (5.1)$$

$$\text{Hassasiyet} = DP / (DP + YN) \quad (5.2)$$

$$\text{Özgüllük} = DN / (DN + YP) \quad (5.3)$$

Dördüncü bölümde bir iznin veya izin grubunun kötücül uygulamalarda bulunma oranının Eş. 4.1'e, iyicil uygulamalarda bulunma oranının Eş. 4.2'ye ve risk puanının Eş. 4.3'e göre hesaplandığı açıklanmıştır. Bu eşitliklere göre kötücül uygulamalarda bir iznin/izin grubunun talep edilme/kullanım oranının artması ile birlikte risk puanı da artmakta, buna karşın iyicil uygulamalarda bahse konu iznin/izin grubunun talep edilme/kullanım oranının artması ile birlikte risk puanı azalmaktadır. Bu nedenle toplam risk puanı arttıkça analiz edilen uygulamanın kötücül olma ihtimali de yükselmektedir. Buna bağlı olarak eşik değer ne kadar düşük olursa kötücül yazılımların kapsanma ihtimali o kadar yüksek olmakta ve dolayısıyla hassasiyet artmaktadır. Buna karşın eşik değerin azalması ile birlikte iyicil uygulamaların kapsanma ihtimali de azalmakta ve dolayısıyla özgüllük de azalmaktadır. Bu tez çalışması kapsamında analiz edilen tüm iyicil ve kötücül uygulamaların puanlandırılması tamamlandıktan sonra iyicil ve kötücül uygulama grupları sırasıyla tüm puan türlerinde ayrı ayrı karşılaştırılmıştır. Karşılaştırmalarda hassasiyetin ve özgüllüğün %90'ın üzerine çıktığı

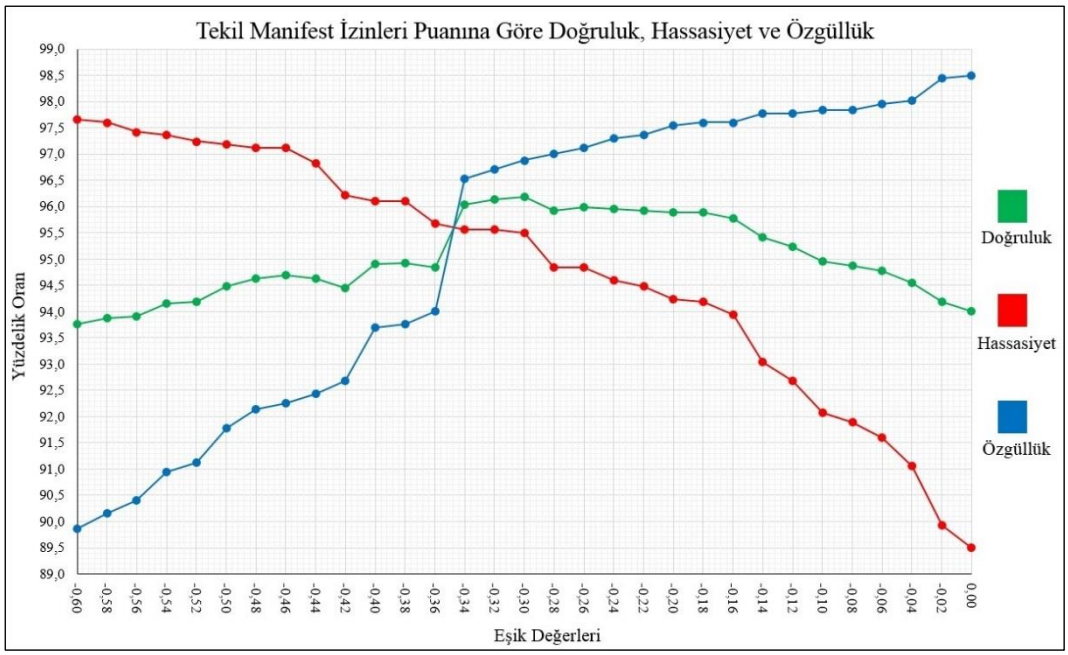
değerler arasındaki aralık dikkate alınmış ve bu aralıkta en yüksek doğruluğa ulaşılan eşik değer ya da değerler bulunmuştur.

Buna göre tekil manifest izinleri puanı için 0,00 ile -0,60 arasında 0,02 puanlık aralıklarla değişen eşik değerlerine göre iyicil ve kötücül uygulamaların ayrımı belirlenmiştir. Tekil manifest izinleri puanları eşik değerlerinin altında kalan iyicil uygulamaların sayısı (doğru-negatif, özgüllük) ile puanları eşik değerlere eşit olan ya da bunların üzerinde kalan kötücül uygulamaların sayısını (doğru-pozitif, hassasiyet) gösteren analiz sonuçları Çizelge 5.13'te sunulmuştur.

Çizelge 5.13. Tekil manifest izinleri puanı analiz sonuçları

Eşik Değeri (Puan)	İyicil Uygulamalar		Kötücül Uygulamalar		Doğruluk
	Adet	Yüzde	Adet	Yüzde	Yüzde
0,00	1641	%98,50	1491	%89,50	%94,00
-0,02	1640	%98,44	1498	%89,92	%94,18
-0,04	1633	%98,02	1517	%91,06	%94,54
-0,06	1632	%97,96	1526	%91,60	%94,78
-0,08	1630	%97,84	1531	%91,90	%94,87
-0,10	1630	%97,84	1534	%92,08	%94,96
-0,12	1629	%97,78	1544	%92,68	%95,23
-0,14	1629	%97,78	1550	%93,04	%95,41
-0,16	1626	%97,60	1565	%93,94	%95,77
-0,18	1626	%97,60	1569	%94,18	%95,89
-0,20	1625	%97,54	1570	%94,24	%95,89
-0,22	1622	%97,36	1574	%94,48	%95,92
-0,24	1621	%97,30	1576	%94,60	%95,95
-0,26	1618	%97,12	1580	%94,84	%95,98
-0,28	1616	%97,00	1580	%94,84	%95,92
-0,30	1614	%96,88	1591	%95,50	%96,19
-0,32	1611	%96,70	1592	%95,56	%96,13
-0,34	1608	%96,52	1592	%95,56	%96,04
-0,36	1566	%94,00	1594	%95,68	%94,84
-0,38	1562	%93,76	1601	%96,10	%94,93
-0,40	1561	%93,70	1601	%96,10	%94,90
-0,42	1544	%92,68	1603	%96,22	%94,45
-0,44	1540	%92,44	1613	%96,82	%94,63
-0,46	1537	%92,26	1618	%97,12	%94,69
-0,48	1535	%92,14	1618	%97,12	%94,63
-0,50	1529	%91,78	1619	%97,18	%94,48
-0,52	1518	%91,12	1620	%97,24	%94,18
-0,54	1515	%90,94	1622	%97,36	%94,15
-0,56	1506	%90,40	1623	%97,42	%93,91
-0,58	1502	%90,16	1626	%97,60	%93,88
-0,60	1497	%89,86	1627	%97,66	%93,76

Çizelgede yer alan sonuçlara göre doğruluk, hassasiyet ve özgüllüğün değişimi Şekil 5.2'de gösterilmiştir. Buna göre eşik değerinin -0,30 olduğu durumda en yüksek doğruluk seviyesine (%96,19) ulaşılmış olup bu eşik değerinde hassasiyet %95,50, özgüllük %96,88 olmuştur.



Şekil 5.2. Tekil manifest izinleri puanına göre doğruluk, hassasiyet ve özgüllük

Tekil kod izinleri puanı için 0,05 ile -1,30 arasında 0,05 puanlık aralıklarla değişen eşik değerlerine göre iyicil ve kötücül uygulamaların ayrımı belirlenmiştir. Tekil kod izinleri puanları eşik değerlerinin altında kalan iyicil uygulamaların sayısı (doğru-negatif, özgüllük) ile puanları eşik değerlere eşit olan ya da bunların üzerinde kalan kötücül uygulamaların sayısını (doğru-pozitif, hassasiyet) gösteren analiz sonuçları Çizelge 5.14'te sunulmuştur.

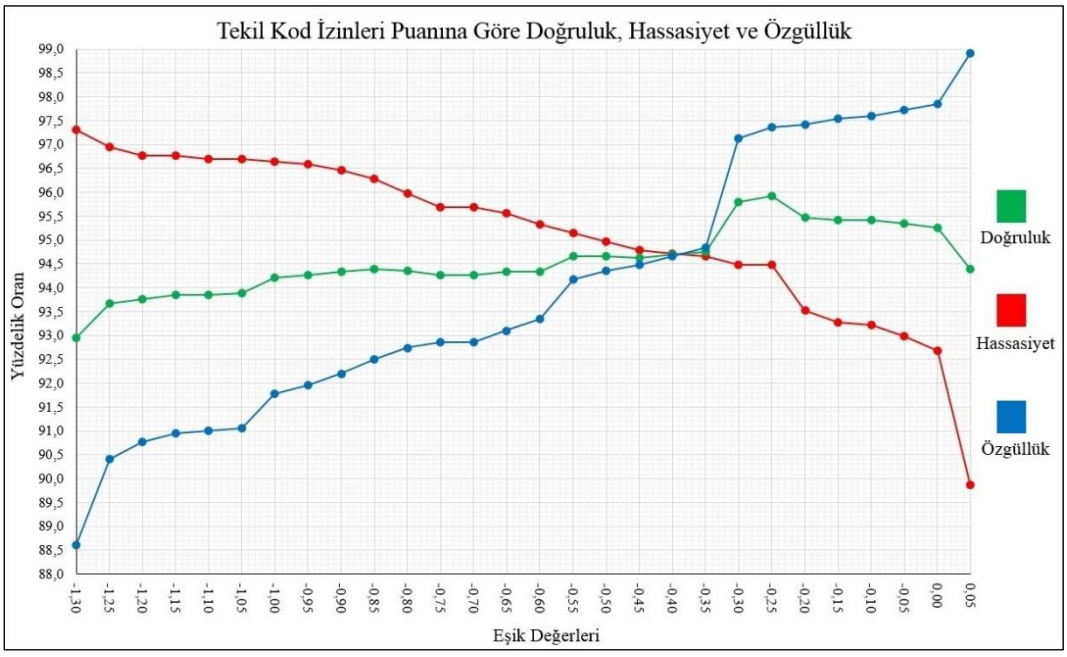
Çizelge 5.14. Tekil kod izinleri puanı analiz sonuçları

Eşik Değeri (Puan)	İyicil Uygulamalar		Kötücül Uygulamalar		Doğruluk
	Adet	Yüzde	Adet	Yüzde	Yüzde
0,05	1648	%98,92	1497	%89,86	%94,39
0,00	1630	%97,84	1544	%92,68	%95,26
-0,05	1628	%97,72	1549	%92,98	%95,35
-0,10	1626	%97,60	1553	%93,22	%95,41
-0,15	1625	%97,54	1554	%93,28	%95,41
-0,20	1623	%97,42	1558	%93,52	%95,47
-0,25	1622	%97,36	1574	%94,48	%95,92

Çizelge 5.14. (devam) Tekil kod izinleri puanı analiz sonuçları

-0,30	1618	%97,12	1574	%94,48	%95,80
-0,35	1580	%94,84	1577	%94,66	%94,75
-0,40	1577	%94,66	1578	%94,72	%94,69
-0,45	1574	%94,48	1579	%94,78	%94,63
-0,50	1572	%94,36	1582	%94,96	%94,66
-0,55	1569	%94,18	1585	%95,14	%94,66
-0,60	1555	%93,34	1588	%95,32	%94,33
-0,65	1551	%93,10	1592	%95,56	%94,33
-0,70	1547	%92,86	1594	%95,68	%94,27
-0,75	1547	%92,86	1594	%95,68	%94,27
-0,80	1545	%92,74	1599	%95,98	%94,36
-0,85	1541	%92,50	1604	%96,28	%94,39
-0,90	1536	%92,20	1607	%96,46	%94,33
-0,95	1532	%91,96	1609	%96,58	%94,27
-1,00	1529	%91,78	1610	%96,64	%94,21
-1,05	1517	%91,06	1611	%96,70	%93,88
-1,10	1516	%91,00	1611	%96,70	%93,85
-1,15	1515	%90,94	1612	%96,76	%93,85
-1,20	1512	%90,76	1612	%96,76	%93,76
-1,25	1506	%90,40	1615	%96,94	%93,67
-1,30	1476	%88,60	1621	%97,30	%92,95

Çizelgede yer alan sonuçlara göre doğruluk, hassasiyet ve özgüllüğün değişimi Şekil 5.3'te gösterilmiştir. Buna göre eşik değerinin -0,25 olduğu durumda en yüksek doğruluk seviyesine (%95,92) ulaşılmış olup bu eşik değerinde hassasiyet %94,48, özgüllük %97,36 olmuştur.



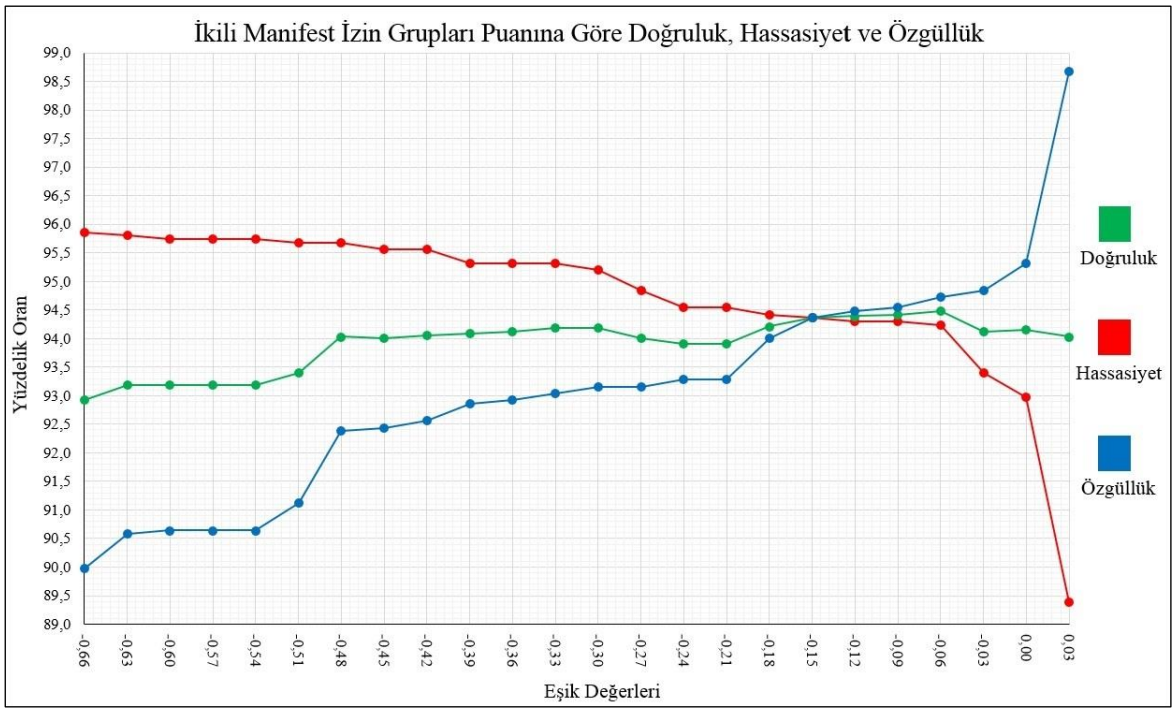
Şekil 5.3. Tekil kod izinleri puanına göre doğruluk, hassasiyet ve özgüllük

İkili manifest izin grupları puanı için 0,03 ile -0,66 arasında 0,03 puanlık aralıklarla değişen eşik değerlerine göre iyicil ve kötücül uygulamaların ayrımı belirlenmiştir. İkili manifest izin grupları puanları eşik değerlerinin altında kalan iyicil uygulamaların sayısı (doğru-negatif, özgüllük) ile puanları eşik değerlere eşit olan ya da bunların üzerinde kalan kötücül uygulamaların sayısını (doğru-pozitif, hassasiyet) gösteren analiz sonuçları Çizelge 5.15'te sunulmuştur.

Çizelge 5.15. İkili manifest izin grupları puanı analiz sonuçları

Eşik Değeri (Puan)	İyicil Uygulamalar		Kötücül Uygulamalar		Doğruluk
	Adet	Yüzde	Adet	Yüzde	Yüzde
0,03	1644	%98,68	1489	%89,38	%94,03
0,00	1588	%95,32	1549	%92,98	%94,15
-0,03	1580	%94,84	1556	%93,40	%94,12
-0,06	1578	%94,72	1570	%94,24	%94,48
-0,09	1575	%94,54	1571	%94,30	%94,42
-0,12	1574	%94,48	1571	%94,30	%94,39
-0,15	1572	%94,36	1572	%94,36	%94,36
-0,18	1566	%94,00	1573	%94,42	%94,21
-0,21	1554	%93,28	1575	%94,54	%93,91
-0,24	1554	%93,28	1575	%94,54	%93,91
-0,27	1552	%93,16	1580	%94,84	%94,00
-0,30	1552	%93,16	1586	%95,20	%94,18
-0,33	1550	%93,04	1588	%95,32	%94,18
-0,36	1548	%92,92	1588	%95,32	%94,12
-0,39	1547	%92,86	1588	%95,32	%94,09
-0,42	1542	%92,56	1592	%95,56	%94,06
-0,45	1540	%92,44	1592	%95,56	%94,00
-0,48	1539	%92,38	1594	%95,68	%94,03
-0,51	1518	%91,12	1594	%95,68	%93,40
-0,54	1510	%90,64	1595	%95,74	%93,19
-0,57	1510	%90,64	1595	%95,74	%93,19
-0,60	1510	%90,64	1595	%95,74	%93,19
-0,63	1509	%90,58	1596	%95,80	%93,19
-0,66	1499	%89,98	1597	%95,86	%92,92

Çizelgede yer alan sonuçlara göre doğruluk, hassasiyet ve özgüllüğün değişimi Şekil 5.4'te gösterilmiştir. Buna göre eşik değerinin -0,06 olduğu durumda en yüksek doğruluk seviyesine (%94,48) ulaşılmış olup bu eşik değerinde hassasiyet %94,24, özgüllük %94,72 olmuştur.



Şekil 5.4. İkili manifest izin grupları puanına göre doğruluk, hassasiyet ve özgüllük

İkili kod izin grupları puanı için -3,50 ile -5,30 arasında 0,06 puanlık aralıklarla değişen eşik değerlerine göre iyicil ve kötücül uygulamaların ayrımı belirlenmiştir. İkili kod izin grupları puanları eşik değerlerinin altında kalan iyicil uygulamaların sayısı (doğru-negatif, özgüllük) ile puanları eşik değerlere eşit olan ya da bunların üzerinde kalan kötücül uygulamaların sayısını (doğru-pozitif, hassasiyet) gösteren analiz sonuçları Çizelge 5.16'da sunulmuştur.

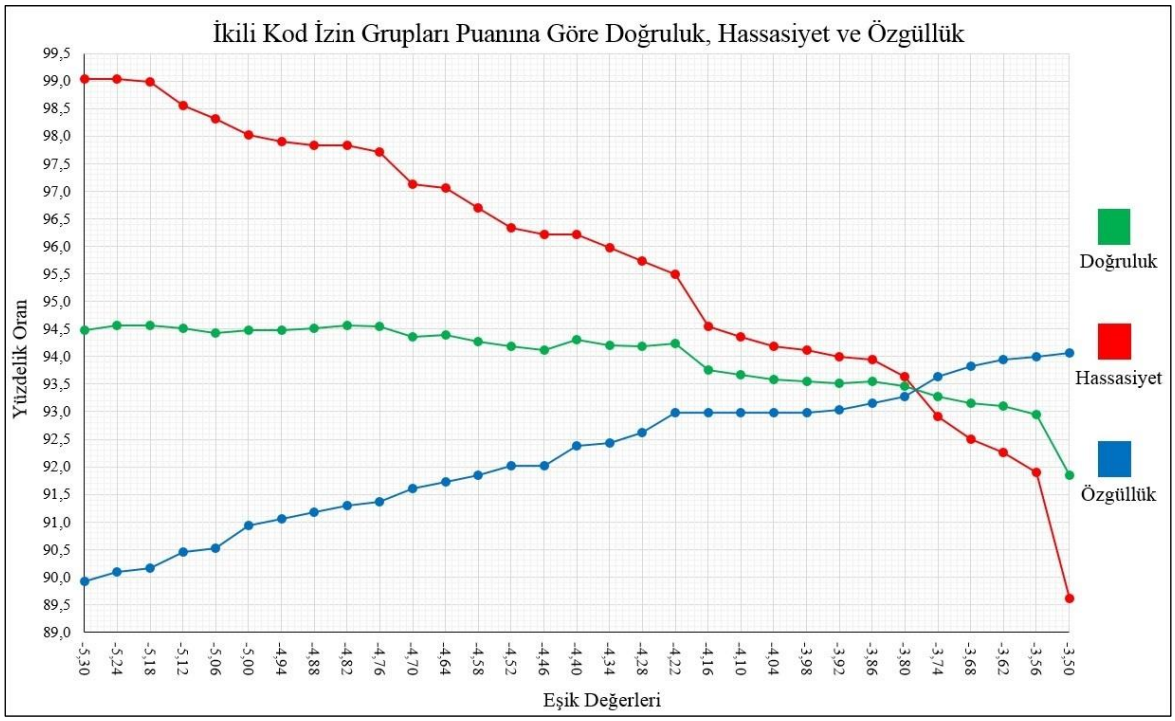
Çizelge 5.16. İkili kod izin grupları puanı analiz sonuçları

Eşik Değeri (Puan)	İyicil Uygulamalar		Kötücül Uygulamalar		Doğruluk
	Adet	Yüzde	Adet	Yüzde	Yüzde
-3,50	1567	%94,06	1493	%89,62	%91,84
-3,56	1566	%94,00	1531	%91,90	%92,95
-3,62	1565	%93,94	1537	%92,26	%93,10
-3,68	1563	%93,82	1541	%92,50	%93,16
-3,74	1560	%93,64	1548	%92,92	%93,28
-3,80	1554	%93,28	1560	%93,64	%93,46
-3,86	1552	%93,16	1565	%93,94	%93,55
-3,92	1550	%93,04	1566	%94,00	%93,52
-3,98	1549	%92,98	1568	%94,12	%93,55
-4,04	1549	%92,98	1569	%94,18	%93,58
-4,10	1549	%92,98	1572	%94,36	%93,67
-4,16	1549	%92,98	1575	%94,54	%93,76
-4,22	1549	%92,98	1591	%95,50	%94,24

Çizelge 5.16. (devam) İkili kod izin grupları puanı analiz sonuçları

-4,28	1543	%92,62	1595	%95,74	%94,18
-4,34	1540	%92,44	1599	%95,98	%94,21
-4,40	1539	%92,38	1603	%96,22	%94,30
-4,46	1533	%92,02	1603	%96,22	%94,12
-4,52	1533	%92,02	1605	%96,34	%94,18
-4,58	1530	%91,84	1611	%96,70	%94,27
-4,64	1528	%91,72	1617	%97,06	%94,39
-4,70	1526	%91,60	1618	%97,12	%94,36
-4,76	1522	%91,36	1628	%97,72	%94,54
-4,82	1521	%91,30	1630	%97,84	%94,57
-4,88	1519	%91,18	1630	%97,84	%94,51
-4,94	1517	%91,06	1631	%97,90	%94,48
-5,00	1515	%90,94	1633	%98,02	%94,48
-5,06	1508	%90,52	1638	%98,32	%94,42
-5,12	1507	%90,46	1642	%98,56	%94,51
-5,18	1502	%90,16	1649	%98,98	%94,57
-5,24	1501	%90,10	1650	%99,04	%94,57
-5,30	1498	%89,92	1650	%99,04	%94,48

Çizelgede yer alan sonuçlara göre doğruluk, hassasiyet ve özgüllüğün değişimi Şekil 5.5'te gösterilmiştir. Buna göre eşik değerinin -4,82, -5,18 ve -5,24 olduğu durumlarda en yüksek doğruluk seviyesine (%94,57) ulaşılmış olup bu eşik değerlerinde hassasiyet sırasıyla %97,84, %98,98 ve %99,04, özgüllük sırasıyla %91,30, %90,16 ve %90,10 olmuştur.



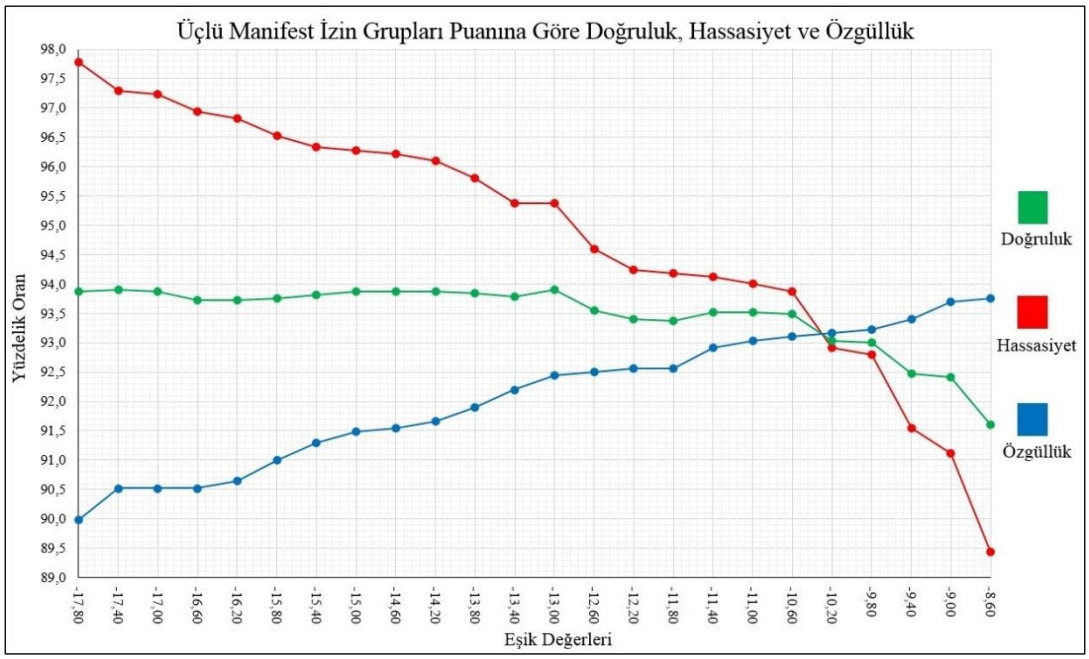
Şekil 5.5. İkili kod izin grupları puanına göre doğruluk, hassasiyet ve özgüllük

Üçlü manifest izin grupları puanı için 0,03 ile -0,66 arasında 0,03 puanlık aralıklarla değişen eşik değerlerine göre iyicil ve kötücül uygulamaların ayrımı belirlenmiştir. Üçlü manifest izin grupları puanları eşik değerlerinin altında kalan iyicil uygulamaların sayısı (doğru-negatif, özgüllük) ile puanları eşik değerlere eşit olan ya da bunların üzerinde kalan kötücül uygulamaların sayısını (doğru-pozitif, hassasiyet) gösteren analiz sonuçları Çizelge 5.17'de sunulmuştur.

Çizelge 5.17. Üçlü manifest izin grupları puanı analiz sonuçları

Eşik Değeri (Puan)	İyicil Uygulamalar		Kötücül Uygulamalar		Doğruluk
	Adet	Yüzde	Adet	Yüzde	Yüzde
-8,60	1562	%93,76	1490	%89,44	%91,60
-9,00	1561	%93,70	1518	%91,12	%92,41
-9,40	1556	%93,40	1525	%91,54	%92,47
-9,80	1553	%93,22	1546	%92,80	%93,01
-10,20	1552	%93,16	1548	%92,92	%93,04
-10,60	1551	%93,10	1564	%93,88	%93,49
-11,00	1550	%93,04	1566	%94,00	%93,52
-11,40	1548	%92,92	1568	%94,12	%93,52
-11,80	1542	%92,56	1569	%94,18	%93,37
-12,20	1542	%92,56	1570	%94,24	%93,40
-12,60	1541	%92,50	1576	%94,60	%93,55
-13,00	1540	%92,44	1589	%95,38	%93,91
-13,40	1536	%92,20	1589	%95,38	%93,79
-13,80	1531	%91,90	1596	%95,80	%93,85
-14,20	1527	%91,66	1601	%96,10	%93,88
-14,60	1525	%91,54	1603	%96,22	%93,88
-15,00	1524	%91,48	1604	%96,28	%93,88
-15,40	1521	%91,30	1605	%96,34	%93,82
-15,80	1516	%91,00	1608	%96,52	%93,76
-16,20	1510	%90,64	1613	%96,82	%93,73
-16,60	1508	%90,52	1615	%96,94	%93,73
-17,00	1508	%90,52	1620	%97,24	%93,88
-17,40	1508	%90,52	1621	%97,30	%93,91
-17,80	1499	%89,98	1629	%97,78	%93,88

Çizelgede yer alan sonuçlara göre doğruluk, hassasiyet ve özgüllüğün değişimi Şekil 5.6'da gösterilmiştir. Buna göre eşik değerinin -13,0 ve -17,4 olduğu durumlarda en yüksek doğruluk seviyesine (%93,91) ulaşılmış olup bu eşik değerlerinde hassasiyet sırasıyla %95,38 ve %97,30, özgüllük sırasıyla %92,44 ve %90,52 olmuştur.



Şekil 5.6. Üçlü manifest izin grupları puanına göre doğruluk, hassasiyet ve özgüllük

Üçlü kod izin grupları puanı için -10 ile -34 arasında 1 puanlık aralıklarla değişen eşik değerlerine göre iyicil ve kötücül uygulamaların ayrımı belirlenmiştir. Üçlü kod izin grupları puanları eşik değerlerinin altında kalan iyicil uygulamaların sayısı (doğru-negatif, özgüllük) ile puanları eşik değerlere eşit olan ya da bunların üzerinde kalan kötücül uygulamaların sayısını (doğru-pozitif, hassasiyet) gösteren analiz sonuçları Çizelge 5.18'de sunulmuştur.

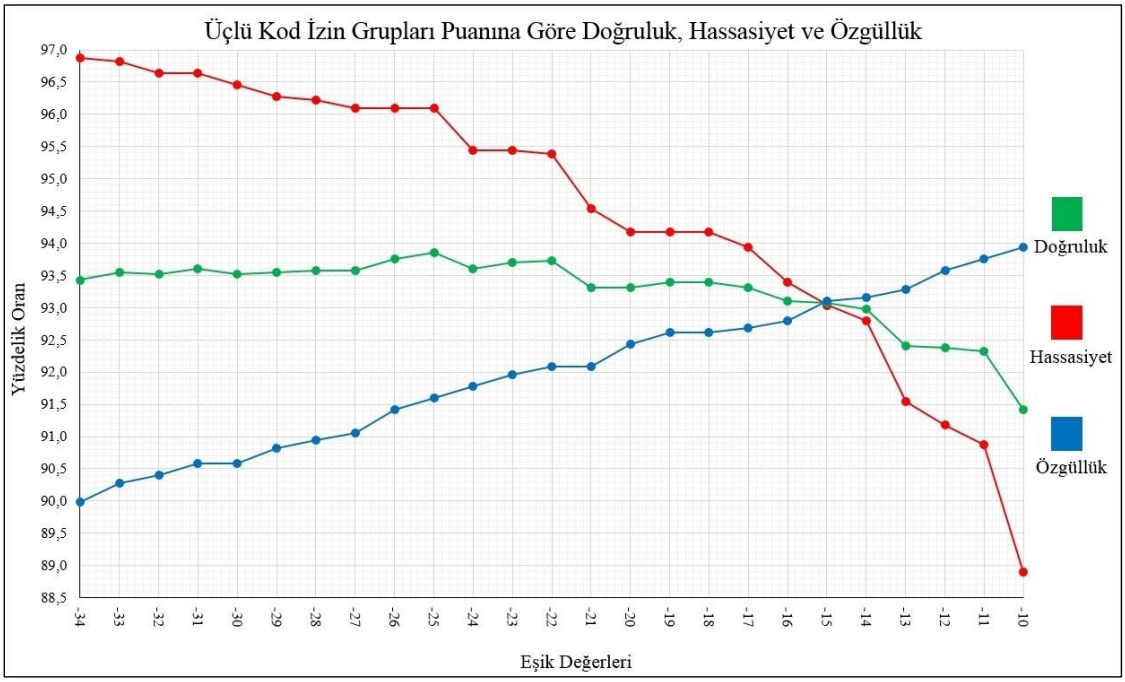
Çizelge 5.18. Üçlü kod izin grupları puanı analiz sonuçları

Eşik Değeri (Puan)	İyicil Uygulamalar		Kötücül Uygulamalar		Doğruluk
	Adet	Yüzde	Adet	Yüzde	Yüzde
-10	1565	%93,94	1481	%88,90	%91,42
-11	1562	%93,76	1514	%90,88	%92,32
-12	1559	%93,58	1519	%91,18	%92,38
-13	1554	%93,28	1525	%91,54	%92,41
-14	1552	%93,16	1546	%92,80	%92,98
-15	1551	%93,10	1550	%93,04	%93,07
-16	1546	%92,80	1556	%93,40	%93,10
-17	1544	%92,68	1565	%93,94	%93,31
-18	1543	%92,62	1569	%94,18	%93,40
-19	1543	%92,62	1569	%94,18	%93,40
-20	1540	%92,44	1569	%94,18	%93,31
-21	1534	%92,08	1575	%94,54	%93,31
-22	1534	%92,08	1589	%95,38	%93,73
-23	1532	%91,96	1590	%95,44	%93,70

Çizelge 5.18. (devam) Üçlü kod izin grupları puanı analiz sonuçları

-24	1529	%91,78	1590	%95,44	%93,61
-25	1526	%91,60	1601	%96,10	%93,85
-26	1523	%91,42	1601	%96,10	%93,76
-27	1517	%91,06	1601	%96,10	%93,58
-28	1515	%90,94	1603	%96,22	%93,58
-29	1513	%90,82	1604	%96,28	%93,55
-30	1509	%90,58	1607	%96,46	%93,52
-31	1509	%90,58	1610	%96,64	%93,61
-32	1506	%90,40	1610	%96,64	%93,52
-33	1504	%90,28	1613	%96,82	%93,55
-34	1499	%89,98	1614	%96,88	%93,43

Çizelgede yer alan sonuçlara göre doğruluk, hassasiyet ve özgüllüğün değişimi Şekil 5.7'de gösterilmiştir. Buna göre eşik değerinin -25 olduğu durumda en yüksek doğruluk seviyesine (%93,85) ulaşılmış olup bu eşik değerinde hassasiyet %96,10, özgüllük%91,60 olmuştur.



Şekil 5.7. Üçlü kod izin grupları puanına göre doğruluk, hassasiyet ve özgüllük

ANUAS'ın altı puan türüne göre doğruluk, hassasiyet ve özgüllük performanslarının karşılaştırması Çizelge 5.19'da sunulmuştur. Çizelgede TMP tekil manifest izinleri puanını, TKP tekil kod izinleri puanını, İMP ikili manifest izin grupları puanını, İKP ikili kod izin grupları puanını, ÜMP üçlü manifest izin grupları puanını ve ÜKP üçlü kod izin grupları puanını ifade etmektedir.

Çizelge 5.19. Puan türlerine göre doğruluk, hassasiyet ve özgüllük karşılaştırması

		TMP	TKP	İMP	İKP	ÜMP	ÜKP
En yüksek doğruluk		%96,19	%95,92	%94,48	%94,57	%93,91	%93,85
En yüksek doğrulukta hassasiyet		%95,50	%94,48	%94,24	%97,84 %98,98 %99,04	%95,38 %97,30	%96,10
En yüksek doğrulukta özgüllük		%96,88	%97,36	%94,72	%91,30 %90,16 %90,10	%92,44 %90,52	%91,60
Özgüllüğün en az %90 kabul edilmesi halinde	Doğruluk	%93,88	%93,67	%93,19	%94,57	%93,91	%93,55
	Hassasiyet	%97,60	%96,94	%95,80	%99,04	%97,30	%96,82
	Özgüllük	%90,16	%90,40	%90,58	%90,10	%90,52	%90,28

Çizelgeye göre ANUAS'ın analiz aşamasında en yüksek doğruluk performansı %96,19 oranıyla tekil manifest izinleri puanına göre değerlendirme yapıldığında elde edilmiştir. Bu doğruluk oranında %95,50 hassasiyet ve %96,88 özgüllük performansına ulaşılabilmiştir. Altı puan türü arasında sonraki en yüksek doğruluk performansı ise %95,92 oranıyla tekil kod izinleri puanına göre değerlendirme yapıldığında elde edilmiştir. Bu doğruluk oranında %94,48 hassasiyet ve %97,36 özgüllük performansına ulaşılabilmiştir. Bu durum uygulamaların kod izinlerinin incelenmesi neticesinde sistemin en yüksek doğruluk performansına çok yakın bir doğruluk performansı elde edilebildiğini ve iyicil uygulamalar ile kötücül uygulamaların ayrılmasında kod izinlerinin de faydalı olabileceğini göstermektedir. İkili ve üçlü izin gruplarının doğruluk performansına bakıldığında ise oranların tekil izin puanlarıyla olduğu kadar yüksek olmamasına karşın yine de %93,85 ile %94,57 aralığında olduğu ve oldukça yüksek bir performans sergilendiği görülebilmektedir.

Karmaşıklık matrisine göre hassasiyet ile özgüllük arasında bir denge bulunmaktadır. Hassasiyet arttıkça özgüllük azalmakta, yani kötücül uygulamaları doğru tespit etme oranı arttıkça iyicil uygulamaları yanlışlıkla kötücül olarak tespit etme ihtimali de artmaktadır. Tersi durumda özgüllük arttıkça hassasiyet azalmakta, yani iyicil uygulamaları doğru tespit etme oranı arttıkça kötücül uygulamaları yanlışlıkla iyicil olarak tespit ederek gözden kaçırma ihtimali de artmaktadır. Buna bağlı olarak ANUAS'ta özgüllüğün %90'a kadar düşürülmesi, yani analiz edilen her on iyicil uygulamadan birinin yanlışlıkla kötücül olarak sınıflandırılması ihtimali göze alınması halinde hassasiyet oranlarının durumu da incelenmiştir. Buna göre özgüllük en az %90 olarak kabul edildiğinde en yüksek hassasiyet performansı %99,04 oranıyla ikili manifest izin grupları puanına göre değerlendirme

yapıldığında elde edilmiştir. Bu hassasiyet oranında %94,57 doğruluk ve %90,10 özgüllük performansına ulaşılabilmiştir. Bu durum özgüllük oranına bağlı olarak izin gruplarından faydalanmanın sistemin hassasiyet performansını yükseltebildiğini göstermektedir. Özgüllüğün en az %90 olarak kabul edilmesi durumunda diğer beş puan türünün hassasiyet performansları %95,80 ile %97,60 aralığında, doğruluk performansları ise %93,19 ile %93,91 aralığında değişmektedir.

Bu tez çalışması kapsamında geliştirilen ANUAS ile Android işletim sistemine yönelik kötücül uygulamaları statik analiz yöntemlerini kullanarak tespit etmeye dayalı diğer çalışmaların karşılaştırması Çizelge 5.20'de sunulmuştur.

Çizelge 5.20. ANUAS ile diğer statik analiz çalışmalarının karşılaştırması

Çalışmanın Adı	Çalışmanın Özellikleri	Kullanılan Veri Kümeleri	Çalışmanın Başarımı
Drebin: Effective and Explainable Detection of Android Malware in Your Pocket	Manifest ve kaynak kod dosyalarından uygulamanın çeşitli özelliklerini çıkarma ve bunları bir vektöre yerleştirme	123 453 iyicil uygulama, Drebin veri kümesi (5560 kötücül uygulama)	%1 yanlış pozitif oranıyla %94 kötücül uygulama tespit oranı
APK Auditor: Permission-Based Android Malware Detection System	Talep edilen izinleri çıkarma, merkezi analiz sunucusunda izinlere göre analiz, makine öğrenmesi kullanarak eşik değer belirleme	1853 iyicil uygulama, Drebin ve Android Malware Genome Project veri kümeleri (6909 kötücül uygulama)	%88,28 tespit doğruluk oranı
Android Kötücül Yazılımlar için İzin Tabanlı Tespit Sistemi	Talep edilen izinleri çıkarma, bu izinleri kategorilendirme, Naive Bayes ve KNN algoritmaları ile uygulamaları ayrıştırma	1339 iyicil uygulama, Drebin veri kümesi (5555 kötücül uygulama)	%97,29 kötücül, %77,72 iyicil uygulama tespit oranı
Android Mobil Uygulamalar için İzin Karşılaştırma Tabanlı Kötücül Yazılım Tespiti	Talep edilen izinler ile kod içerisinde kullanılan izinleri çıkarma ve karşılaştırma	25 iyicil uygulama, 50 kötücül uygulama	%97,62 kötücül, %80 iyicil uygulama tespit oranı
AndroDialysis: Analysis of Android Intent Effectiveness in Malware Detection	Talep edilen izinler ile niyet filtrelerini çıkarma, kod içerisindeki açık ve gizli niyet nesnelerini bulma	1846 iyicil uygulama, Drebin veri kümesi (5560 kötücül uygulama)	Sadece izinlerle % 83, sadece niyetlerle %91, ikisi birlikte %95,5 doğru pozitif tespit oranı

Çizelge 5.20. (devam) ANUAS ile diğer statik analiz çalışmalarının karşılaştırması

Detecting Android Malicious Apps and Categorizing Benign Apps with Ensemble of Classifiers	Uygulamalardan Destek Vektör Makineleri kullanılarak belirlenen özellikleri çıkarma, sınıflandırma yöntemlerinin birleşimi kullanılarak uygulamaları sınıflandırma	107 327 iyicil uygulama, 8701 kötücül uygulama	%99,39 kötücül uygulama tespit oranı
Web Tabanlı Android Kötücül Yazılım Tespit Sistemi	Talep edilen izinleri çıkarma, sunucuda izinlere göre analiz, Virustotal'de sorgulama, makine öğrenmesi kullanarak eşik değer belirleme	1173 iyicil uygulama, Drebin veri kümesi (5545 kötücül uygulama)	%97,73 doğru tespit oranı
DroidDet: Effective and robust detection of android malware using static analysis along with rotation forest model	Uygulamaların talep edilen izinler, hassas API çağrıları, izlenen sistem olayları ve izin sayısı-smali boyutu oranı özelliklerini çıkarma, rotasyon ormanı yöntemiyle model oluşturma ve karşılaştırma	1065 iyicil uygulama, 1065 kötücül uygulama	%88,26 doğruluk, %88,40 hassasiyet ve %88,16 duyarlılık oranı
An Android mutation malware detection based on deep learning using visualization of importance from codes	Uygulama kodu içerisinde geçen kelimelerin önem değerlerini belirleme, bu değerlerden uygulama resimlerini oluşturma, derin öğrenme ile eğitme ve test etme	720 iyicil uygulama, 720 kötücül uygulama	%92 doğruluk oranı
Tez Çalışması (İzin ve İzin Gruplarına Dayalı Android Kötücül Yazılım Tespit Sistemi)	Talep edilen izinleri çıkarma, izinlerden izin grupları oluşturma, sunucuda izinlere ve izin gruplarına göre analiz	5554 iyicil uygulama, Drebin veri kümesi (5554 kötücül uygulama)	En yüksek doğruluk performansı %96,19 (%95,50 kötücül uygulama, %96,88 iyicil uygulama tespit oranı) Ayrıca %99,04 kötücül uygulama, %90,10 iyicil uygulama tespit ve %94,57 doğruluk oranı

Çizelge 5.20 incelendiğinde tez çalışması kapsamında geliştirilen izin ve izin gruplarına dayalı Android kötücül yazılım tespit sistemi ANUAS'ın diğer çalışmaların önemli bir bölümünden daha yüksek doğruluk, hassasiyet ve özgüllük sonuçlarına ulaşabildiği görülecektir. Bu durum geliştirilen sistemin kötücül yazılımları tespit etme konusunda literatürdeki diğer çalışmalarla karşılaştırıldığında önemi bir başarım sağladığını göstermektedir.

6. SONUÇ VE ÖNERİLER

Android dünyasındaki kötücül yazılımlarla ilgili pek çok araştırmacı ve güvenlik şirketi tarafından yapılan araştırmalara göre hâlihazırda hem resmi uygulama mağazalarında hem de üçüncü parti uygulama mağazalarında sunulan milyonlarca uygulama arasında çok fazla kötücül yazılım da bulunmakta ve hem kötücül yazılım miktarı hem de kötücül yazılım ailesi sayısı günden güne hızla artmaktadır. Bununla birlikte geliştirilen kötücül yazılımlar sürekli olarak daha karmaşık ve daha yetenekli hale getirilmektedir. Buna karşın Android işletim sistemi izin tabanlı güvenlik politikası uygulamakta, ancak bu politika kullanıcılara büyük sorumluluk yüklemesi nedeniyle güvenliği tam olarak sağlayamamaktadır. Bu nedenle mobil cihazlarda güvenliği sağlamak amacıyla kötücül yazılımları cihazlara yüklenmeden önce tespit eden güvenlik çözümlerine büyük ihtiyaç vardır.

Bu kapsamda bu tez çalışmasında analiz edilen bir uygulamanın kötücül ya da iyicil olduğunu tespit etmek üzere web tabanlı bir sistem olan ANUAS geliştirilmiştir. ANUAS analiz edilecek uygulama tarafından talep edilen izinlerin ve uygulama kodunda kullanılan izinleri çıkarmakta ve bu izinlerden ikili ve üçlü izin grupları oluşturmaktadır. Ardından çıkarılan izinler ile bunlardan oluşturulan izin gruplarının kötücül/iyicil uygulamalarda talep edilme/kullanım oranlarını bulmakta, bu oranları kullanarak her bir iznin/izin grubunun risk puanını hesaplamaktadır. Sonuçta uygulamanın talep ettiği veya kod içerisinde kullandığı tüm izinlerin ve izin gruplarının risk puanlarının toplanmasıyla analiz edilen uygulamanın toplam risk puanını hesaplamakta ve bu puana göre uygulamayı iyicil ya da kötücül olarak sınıflandırmaktadır.

Bu yaklaşımla geliştirilen ANUAS öncelikle 3888 iyicil ve 3888 kötücül uygulama ile eğitilmiş, daha sonra 1666 iyicil ve 1666 kötücül uygulama ile test edilmiştir. Testlerin sonucunda ANUAS'ın tekil manifest izin puanına göre değerlendirme yapıldığında %96,19 oranıyla en yüksek doğruluk performansını gösterdiği tespit edilmiştir. Bu doğruluk oranında %95,50 hassasiyet ve %96,88 özgüllük performansına ulaşılabilmiştir. Bu tez çalışması kapsamında kod izinlerinin iyicil ve kötücül uygulamaların ayrılmasında ne kadar başarılı olabileceği de incelenmiştir. Kod izinleri doğruluk performanslarına bakıldığında en yüksek performans %95,92 oranıyla tekil kod izinleri puanına göre değerlendirme yapıldığında elde edilmiştir. Bu doğruluk oranında %94,48 hassasiyet ve %97,36 özgüllük

performansına ulaşılabilmiştir. Bu doğruluk oranı tekil manifest izinlerine göre değerlendirme yapıldığında elde edilen orana oldukça yakın olup kod izinlerinin de uygulamaların sınıflandırılmasında oldukça etkili olabileceğini göstermektedir.

Bu tez çalışması kapsamında ayrıca manifest ve kod izinlerinden elde edilen iki veya üç elemanlı izin gruplarının iyicil ve kötücül uygulamaların ayrılmasında ne kadar başarılı olabileceği de incelenmiştir. İzin gruplarına göre yapılan değerlendirmelerin doğruluk performanslarına bakıldığında oranların tekil izin puanlarıyla olduğu kadar yüksek olmamasına karşın yine de %93,85 ile %94,57 aralığında olduğu ve oldukça yüksek bir performans sergilendiği görülebilmektedir. Bununla birlikte hassasiyetin arttırılması maksadıyla özgüllüğün %90'a kadar düşürülmesi halinde en yüksek hassasiyet performansı %99,04 oranıyla ikili manifest izin grupları puanına göre değerlendirme yapıldığında elde edilmiştir. Bu hassasiyet oranında %94,57 doğruluk ve %90,10 özgüllük performansına ulaşılabilmiştir. Bu durum özgüllük oranına bağlı olarak izin gruplarından faydalanmanın sistemin hassasiyet performansını yükseltebildiğini ve iyicil uygulamalar ile kötücül uygulamaların sınıflandırılmasında izin gruplarının da faydalı olabileceğini göstermektedir.

Bu tez çalışması kapsamında geliştirilen ANUAS'ta hâlihazırda mobil uygulamaların AndroidManifest.xml dosyalarından sadece izin talepleri çıkarılmaktadır. Gelecekte yapılacak çalışmalar kapsamında öncelikle manifest dosyası içerisinde bulunan amaçlar, donanım talepleri gibi diğer alanların da incelenmesi ve iyicil/kötücül uygulamaların sınıflandırılmasındaki etkinliklerinin araştırılması planlanmaktadır. Ayrıca uygulama kodları içerisindeki izin adları dışında API çağrıları, yöntem isimleri gibi diğer ögeleri de inceleyerek kod analizinin etkinliğinin arttırılmasına çalışılacaktır. Bunların dışında hem izin gruplarıyla elde edilen performansların yükseltilebilme olanakları hem de hesaplanan altı farklı puan türünün ayrı ayrı yerine birlikte değerlendirilmesiyle daha yüksek performanslar elde edilip edilemeyeceği araştırılacaktır. Böylece daha detaylı incelemeler yaparak statik analiz performansının yükseltilmesi hedeflenmektedir.

ANUAS hâlihazırda sadece statik analiz yapabilmektedir ancak günümüzde geliştirilen kötücül yazılımların gittikçe daha karmaşık hale gelmesine bağlı olarak bunların sadece statik analiz ile tespit edilebilmesi gün geçtikçe daha da zorlaşmaktadır. Bu nedenle ANUAS'a dinamik analiz yeteneği de kazandırılması ve böylece hibrit analiz yapabilen bir yapıya kavuşturulması planlanmaktadır. Ayrıca sisteme makine öğrenmesi ve derin öğrenme

tekniklerinin de katılmasıyla sistemin öğrenen ve kendisini sürekli geliştiren bir hale getirilmesi düşünülmektedir. Kötücül uygulamaların tespit edilmesi konusunda çalışma yapacak araştırmacılara bu hususları dikkate alarak statik analiz, dinamik analiz, makine öğrenmesi ve derin öğrenme tekniklerini harmanlayarak analiz yapan sistemler geliştirmeye çalışmaları önerilmektedir.

Son olarak kullanıcıların sistemi kullanımını kolaylaştırmak amacıyla mobil cihazlar üzerine kurulacak ve kullanıcıların cihaz üzerinden ANUAS'a dosya ya da indirme bağlantısı göndermelerini sağlayacak bir mobil uygulama geliştirilmesi planlanmaktadır.

KAYNAKLAR

1. Felt, A.P., Finifter, M., Chin, E., Hanna, S. and Wagner, D. (2011). A survey of mobile malware in the wild. *In Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM '11)*. ACM, New York, NY, USA, 3–14.

2. Zhou, Y., Wang, Z., Zhou, W. and Jiang X. (2012). Hey, You, Get Off of My Market: Detecting Malicious Apps in Official and Alternative Android Markets. *In Proceedings of the 19th Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, California, USA.

3. İnternet: Tehditler ve Korunma Yöntemleri Zararlı Programlar URL: http://www.bilgimikoruyorum.org.tr/?b310_zararli_programlar_ve_korunma_yontemleri. Son Erişim Tarihi: 18.08.2019.

4. İnternet: Cyber attacks on Android devices on the rise. URL: https://www.gdatasoftware.com/blog/2018/11/31255-cyber-attacks-on-android-devices-on-the-rise. Son Erişim Tarihi: 18.08.2019.

5. İnternet: Samani, R. and Davis, G. (2019). McAfee Mobile Threat Report Q1, 2019, *McAfee*. Santa Clara, CA, USA. 1-20 URL: https://www.mcafee.com/enterprise/en-us/assets/ reports/rp-mobile-threat-report-2019.pdf. Son Erişim Tarihi: 18.08.2019.

6. İnternet: Pulse Secure Mobile Threat Center (MTC) (2015). 2015 Mobile Threat Report, *Pulse Secure*. San Jose, CA, USA. 1-20. URL: https://www.pulsesecure.net/download/ pages/2819. Son Erişim Tarihi: 18.08.2019.

7. İnternet: F-Secure (2017). F-Secure State of Cyber Security 2017, *F-Secure*. Helsinki, Finland. 1-77. URL: https://blog.f-secure.com/the-state-of-cyber-security-2017. Son Erişim Tarihi: 18.08.2019.

8. Fang, Z., Han W. and Li Y. (2014). Permission based Android security: Issues and countermeasures. *Computers & Security*, 43, 205–218.

9. İnternet: Android (operating system). URL: https://en.wikipedia.org/wiki/Android_(operating_system). Son Erişim Tarihi: 18.08.2019.

10. İnternet: Industry Leaders Announce Open Platform for Mobile Devices. URL: http://www.openhandsetalliance.com/press_110507.html. Son Erişim Tarihi: 18.08.2019.

11. İnternet: Open Handset Alliance. URL: https://en.wikipedia.org/wiki/Open_Handset_Alliance. Son Erişim Tarihi: 18.08.2019.

12. İnternet: Open Handset Alliance. URL: http://www.openhandsetalliance.com. Son Erişim Tarihi: 18.08.2019.

13. İnternet: Android passes BlackBerry as No. 1 on smartphones. URL: https://money.cnn.com/2011/03/07/technology/android/index.htm. Son Erişim Tarihi: 18.08.2019.

14. İnternet: Global mobile OS market share in sales to end users from 1st quarter 2009 to 2nd quarter 2018. URL: https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems. Son Erişim Tarihi: 18.08.2019.

15. İnternet: Smartphone Market Share. URL: https://www.idc.com/promo/smartphone-market-share/os. Son Erişim Tarihi: 18.08.2019.

16. İnternet: Al-Heeti, A. (7 Mayıs 2019). Android is on over 2.5 billion active devices, URL: https://www.cnet.com/news/android-is-on-over-2-5-billion-active-devices. Son Erişim Tarihi: 18.08.2019.

17. İnternet: Protalinski, E. (7 Mayıs 2019). Android passes 2.5 billion monthly active devices. URL: https://venturebeat.com/2019/05/07/android-passes-2-5-billion-monthly-active-devices. Son Erişim Tarihi: 18.08.2019.

18. İnternet: Protalinski, E. (17 Mayıs 2017). Android passes 2 billion monthly active devices. URL: https://venturebeat.com/2017/05/17/android-passes-2-billion-monthly-active-devices. Son Erişim Tarihi: 18.08.2019.

19. İnternet: Lumb, D., Swider, M. (8 Temmuz 2019). Android Q release date, new features and everything you need to know. URL: https://www.techradar.com/news/android-q. Son Erişim Tarihi: 18.08.2019.

20. İnternet: Android version history. URL: https://en.wikipedia.org/wiki/Android_version_history. Son Erişim Tarihi: 18.08.2019.

21. İnternet: Distribution dashboard. URL: https://developer.android.com/about/dashboards. Son Erişim Tarihi: 18.08.2019.

22. İnternet: Android-System-Architecture.svg. URL: https://commons.wikimedia.org/wiki/File:Android-System-Architecture.svg. Son Erişim Tarihi: 18.08.2019.

23. İnternet: Dalvik (software). URL: https://en.wikipedia.org/wiki/Dalvik_(software). Son Erişim Tarihi: 18.08.2019.

24. İnternet: Linder, B. (15 Ekim 2014). What's new in Android 5.0 Lollipop?. URL: http://liliputing.com/2014/10/whats-new-android-5-0-lollipop.html. Son Erişim Tarihi: 18.08.2019.

25. İnternet: Android Runtime. URL: https://en.wikipedia.org/wiki/Android_Runtime. Son Erişim Tarihi: 18.08.2019.

26. İnternet: Android application package. URL: https://en.wikipedia.org/wiki/Android_application_package. Son Erişim Tarihi: 18.08.2019.

27. İnternet: XAPK File Extension. URL: https://fileinfo.com/extension/xapk. Son Erişim Tarihi: 18.08.2019.

28. İnternet: Azza, N. (14 Eylül 2018). Easily Install the .XAPK File on Android. URL: https://www.digitbin.com/xapk. Son Erişim Tarihi: 18.08.2019.

29. İnternet: Glick, K. (28 Eylül 2015). Support for 100MB APKs on Google Play. URL: https://android-developers.googleblog.com/2015/09/support-for-100mb-apks-on-google-play.html. Son Erişim Tarihi: 18.08.2019.

30. İnternet: APK Expansion Files. URL: https://developer.android.com/google/play/expansion-files. Son Erişim Tarihi: 18.08.2019.

31. İnternet: About Android App Bundles. URL: https://developer.android.com/guide/app-bundle. Son Erişim Tarihi: 18.08.2019.

32. İnternet: Google Play. URL: https://en.wikipedia.org/wiki/Google_Play. Son Erişim Tarihi: 18.08.2019.

33. İnternet: Number of Android apps on Google Play. URL: https://www.appbrain.com/stats/number-of-android-apps. Son Erişim Tarihi: 18.08.2019.

34. İnternet: Number of available applications in the Google Play Store from December 2009 to June 2019. URL: https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store. Son Erişim Tarihi: 18.08.2019.

35. İnternet: 10 best third party app stores for Android and other options too!. URL: https://www.androidauthority.com/best-app-stores-936652. Son Erişim Tarihi: 18.08.2019.

36. İnternet: App Manifest Overview. https://developer.android.com/guide/topics/manifest/manifest-intro.html. Son Erişim Tarihi: 18.08.2019.

37. İnternet: Understanding App Permissions. URL: https://guides.codepath.com/android/Understanding-App-Permissions. Son Erişim Tarihi: 18.08.2019.

38. İnternet: Amadeo, R. (5 Ekim 2015). Android 6.0 Marshmallow, thoroughly reviewed. URL: http://arstechnica.com/gadgets/2015/10/android-6-0-marshmallow-thoroughly-reviewed/5. Son Erişim Tarihi: 18.08.2019.

39. İnternet: Android Studio. URL: https://developer.android.com/studio. Son Erişim Tarihi: 18.08.2019.

40. İnternet: Run apps on the Android Emulator. URL: https://developer.android.com/studio/run/emulator. Son Erişim Tarihi: 18.08.2019.

41. İnternet: Manifest.permission. URL: https://developer.android.com/reference/android/Manifest.permission.html. Son Erişim Tarihi: 18.08.2019.

42. İnternet: Permissions overview. https://developer.android.com/guide/topics/permissions/overview. Son Erişim Tarihi: 18.08.2019.

43. Do, Q., Martini, B. and Choo K.K.R. (2015). Exfiltrating data from Android devices. *Computers & Security*, 48, 74-91.

44. İnternet: Lockheimer, H. (2 Şubat 2012). Android and Security. http://googlemobile.blogspot.com/2012/02/android-and-security.html. Son Erişim Tarihi: 18.08.2019.

45. İnternet: Oberheide, J. (21 Haziran 2012). Dissecting the Android Bouncer. URL: https://jon.oberheide.org/blog/2012/06/21/dissecting-the-android-bouncer. Son Erişim Tarihi: 18.08.2019.

46. İnternet: Hou, O. (20 Temmuz 2012). A Look at Google Bouncer. URL: http://blog.trendmicro.com/trendlabs-security-intelligence/a-look-at-google-bouncer. Son Erişim Tarihi: 18.08.2019.

47. İnternet: BrainTest – A New Level of Sophistication in Mobile Malware. URL: https://blog.checkpoint.com/2015/09/21/braintest-a-new-level-of-sophistication-in-mobile-malware. Son Erişim Tarihi: 18.08.2019.

48. İnternet: Burke, D. (17 Mayıs 2017). Android: celebrating a big milestone together with you. URL: https://www.blog.google/products/android/2bn-milestone. Son Erişim Tarihi: 18.08.2019.

49. İnternet: Cunningham, E. (17 Mayıs 2017). Keeping you safe with Google Play Protect. URL: https://blog.google/products/android/google-play-protect. Son Erişim Tarihi: 18.08.2019.

50. İnternet: Amadeo, R. (4 Eylül 2017). Android 8.0 Oreo, thoroughly reviewed. URL: https://arstechnica.com/gadgets/2017/09/android-8-0-oreo-thoroughly-reviewed/5. Son Erişim Tarihi: 18.08.2019.

51. İnternet: Ahn, A. (30 Ocak 2018). How we fought bad apps and malicious developers in 2017. URL: https://android-developers.googleblog.com/2018/01/how-we-fought-bad-apps-and-malicious.html. Son Erişim Tarihi: 18.08.2019.

52. İnternet: AV-Comparatives Independent Tests of Anti-Virus Software. URL: https://www.av-comparatives.org. Son Erişim Tarihi: 18.08.2019.

53. İnternet: Funding. URL: https://www.av-comparatives.org/funding. Son Erişim Tarihi: 18.08.2019.

54. İnternet: Mobile Security Review 2019. URL: https://www.av-comparatives.org/tests/mobile-security-review-2019. Son Erişim Tarihi: 18.08.2019.

55. İnternet: The AppInChina App Store Index. URL: https://www.appinchina.co/market/app-stores. Son Erişim Tarihi: 18.08.2019.

56. İnternet: Doffman, Z. (1 Ağustos 2019). Android Warning As 32 Million 'Harmful' Apps Downloaded From Google Play In July. URL: https://www.forbes.com/sites/zakdoffman/2019/08/01/android-warning-32-million-installs-of-harmful-apps-from-google-play-in-july. Son Erişim Tarihi: 18.08.2019.

57. İnternet: Doffman, Z. (24 Haziran 2019). Android Warning: Thousands Of Dangerous Copycat Apps On Google Play, Study Finds. URL: https://www.forbes.com/sites/zakdoffman/2019/06/24/google-play-hosts-thousands-of-malware-riddled-copycat-apps-ai-study-finds. Son Erişim Tarihi: 18.08.2019.

58. Felt, A.P., Chin, E., Hanna, S., Song, D. and Wagner D. (2011). Android permissions demystified. *In Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS '11)*. ACM, New York, NY, USA, 627-637.

59. Au, K.W.Y., Zhou, Y.F., Huang, Z. and Lie D. (2012). Pscout: analyzing the android permission specification. *In Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS '12)*. ACM, New York, NY, USA, 217-228.

60. Zhou, Y. and Jiang, X. (2012). Dissecting Android malware: characterization and evolution. *In Proceedings of the 2012 IEEE Symposium on Security and Privacy (SP '12)*. IEEE Computer Society, Washington, DC, USA, 95-109.

61. Arp, D., Spreitzenbarth, M., Hübner M., Gascon, H. and Rieck K. (2014). Drebin: Effective and Explainable Detection of Android Malware in Your Pocket. *In Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, California, USA.

62. Moonsamy, V., Rong, J. and Liu, S. (2014). Mining permission patterns for contrasting clean and malicious android applications. *Future Generation Computer Systems*, 36, 122-132.

63. Kabakuş, A.T., Doğru, İ.A. and Çetin, A. (2015). APK Auditor: Permission-based Android malware detection system. *Digital Investigation*, 13, 1-14.

64. Utlu, A. ve Doğru, İ.A. (2017). Android kötücül yazılımlar için izin tabanlı tespit sistemi. *Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi*, 32:(4), 1015-1024.

65. Arslan, R.S., Doğru, İ.A. ve Barışçı, N. (2017). Android mobil uygulamalar için izin karşılaştırma tabanlı kötücül yazılım tespiti. *Politeknik Dergisi*, 20(1), 175-189.

66. Sokolova, K., Perez, C. and Lemercier, M. (2017). Android application classification and anomaly detection with graph-based permission patterns. *Decision Support Systems*, 93, 62-76.

67. Feizollah, A., Anuar, N. B., Salleh, R., Suarez-Tangil, G. and Furnell, S. (2017). AndroDialysis: Analysis of Android Intent Effectiveness in Malware Detection. *Computers & Security*, 65, 121-134.

68. Wang, W., Li, Y., Wang, X., Liu, J. and Zhang, X. (2018). Detecting android malicious apps and categorizing benign apps with ensemble of classifiers. *Future Generation Computer Systems*, 78, 987-994.

69. Kiraz, Ö. (2018). *Web Tabanlı Android Kötücül Yazılım Tespit Sistemi*, Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

70. Zhu, H.-J., You, Z.-H., Zhu, Z.-X., Shi, W.-L., Chen, X., Cheng, L. (2018). DroidDet: Effective and robust detection of android malware using static analysis along with rotation forest model. *Neurocomputing*, 272, 638–646.

71. İnternet: VirusShare.com - Because Sharing is Caring. URL: https://virusshare.com. Son Erişim Tarihi: 18.08.2019.

72. Yen, Y.-S. and Sun, H.-M. (2019). An Android mutation malware detection based on deep learning using visualization of importance from codes. *Microelectronics Reliability*, 93, 109–114.

73. İnternet: Caffe – Deep Learning Framework by BAIR. URL: https://caffe. berkeleyvision.org. Son Erişim Tarihi: 18.08.2019.

74. Enck, W., Gilbert, P., Chun, B.G., Cox, L.P., Jung, J., McDaniel, P. and Sheth, A.N. (2010). TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones. *In Proceedings of the 9th USENIX conference on Operating systems design and implementation (OSDI'10)*. USENIX Association, Berkeley, CA, USA, 1-6.

75. Beresford, A.R., Rice, A., Skehin, N. and Sohan, R. (2011). Mockdroid: trading privacy for application functionality on smartphones. *In Proceedings of the 12th Workshop on Mobile Computing Systems and Applications (HotMobile '11)*. ACM, New York, NY, USA, 49-54.

76. Hornyack, P., Han, S., Jung, J., Schechter, S. and Wetherall, D. (2011). These aren't the droids you're looking for: retrofitting android to protect data from imperious applications, *In Proceedings of the 18th ACM conference on Computer and communications security (CCS '11)*. ACM, New York, NY, USA, 639-652.

77. Zhou, Y., Zhang, X., Jiang, X. and Freeh V.W. (2011). Taming information-stealing smartphone applications (on android). In McCune, J.M., Balacheff, B., Perrig, A., Sadeghi, A.-R. and Sasse A. (Eds.), *In Proceedings of the 4th International Conference on Trust and Trustworthy Computing (TRUST'11)*. Springer-Verlag, Berlin, Heidelberg, Germany, 93-107.

78. Bugiel, S., Davi, L., Dmitrienko, A., Fischer, T., Sadeghi, A. (2011). XManDroid: a new Android evolution to mitigate privilege escalation attacks; Technical Report TR-2011-04. *Technische Universität Darmstadt*, Darmstadt, Germany, 1-17.

79. Shabtai, A., Tenenboim-Chekina, L., Mimran, D., Rokach, L., Shapira, B. and Elovici, Y. (2014). Mobile malware detection through analysis of deviations in application network behavior. *Computers & Security*, 43, 1-18.

80. Jang, J.-W., Yun, J., Mohaisen, A., Woo, J. and Kim, H.K. (2016). Detecting and classifying method based on similarity matching of Android malware behavior with profile. *SpringerPlus*, 5:273, 1-23.

81. İnternet: Droidbox - Android Application Sandbox. URL: https://code.google.com/ archive/p/droidbox. Son Erişim Tarihi: 18.08.2019.

82. Heuser, S., Negro, M., Pendyala, P.K. and Sadeghi, A.-R. (2017). DroidAuditor: Forensic Analysis of Application-Layer Privilege Escalation Attacks on Android (Short Paper). In Grossklags, J. and Preneel, B. (Eds.), *FC 2016: Financial Cryptography and Data Security*. Lecture Notes in Computer Science, vol 9603. Springer, Berlin, Heidelberg, Germany, 260-268.

83. Heuser, S., Nadkarni, A., Enck, W., Sadeghi, A.-R. (2014). ASM: A Programmable Interface for Extending Android Security. *In Proceedings of the 23rd USENIX Conference on Security Symposium (SEC '14)*. USENIX Association, Berkeley, CA, USA, 1005-1019.

84. Spreitzenbarth, M., Freiling, F.C., Echtler, F., Schreck, T. and Hoffmann, J. (2013). Mobilesandbox: Having a Deeper Look into Android Applications. *In Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC '13)*. ACM, New York, NY, USA, 1808-1815.

85. Geneiatakis, D., Fovino, I.N., Kounelis, I. and Stirparo, P. (2015). A permission verification approach for android mobile applications. *Computer Systems*, 49, 192–205.

86. Bartel, A., Klein, J., Le Traon, Y., Monperrus, M. (2012). Dexpler: Converting Android Dalvik Bytecode to Jimple for Static Analysis with Soot, *In Proceedings of the ACM SIGPLAN International Workshop on State of the Art in Java Program Analysis (SOAP '12)*. ACM, New York, NY, USA, 27-38.

87. Chang, W.-L., Sun, H.-M. and Wu, W. (2016). An Android Behavior-Based Malware Detection Method using Machine Learning, *In Proceedings of the 2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*. IEEE Computer Society, Washington, DC, USA, 1-4.

88. Kabakuş, A.T. and Doğru, İ.A. (2018). An in-depth analysis of Android malware using hybrid techniques. *Digital Investigation*, 24, 25–33.

89. İnternet: A tool for reverse engineering Android apk files. URL: https://ibotpeaches. github.io/Apktool. Son Erişim Tarihi: 18.08.2019.

90. İnternet: Upadhyay, S. (15 Ocak 2015). What is smali in Android?. URL: https://www.quora.com/What-is-smali-in-Android. Son Erişim Tarihi: 18.08.2019.

91. İnternet: Class HashMap<K,V>. URL: https://docs.oracle.com/javase/8/docs/api/ java/util/HashMap.html. Son Erişim Tarihi: 18.08.2019.

92. İnternet: Prime User Interface. URL: https://www.primefaces.org. Son Erişim Tarihi: 18.08.2019.

93. İnternet: The Drebin Dataset. URL: https://www.sec.cs.tu-bs.de/~danarp/drebin. Son Erişim Tarihi: 18.08.2019.

94. İnternet: About us. URL: https://support.virustotal.com/hc/en-us/categories/ 360000160117-About-us. Son Erişim Tarihi: 18.08.2019.

95. İnternet: How it works. URL: https://support.virustotal.com/hc/en-us/articles/ 115002126889-How-it-works. Son Erişim Tarihi: 18.08.2019.

96. İnternet: Getting started. URL: https://developers.virustotal.com/reference. Son Erişim Tarihi: 18.08.2019.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: ÖNDER, Murat
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 24.04.1983, Bulgaristan
Medeni hali	: Evli
Telefon	: 0 (535) 640 07 89
E-mail	: murat.onder1@gazi.edu.tr muratonder1@gmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet
Yüksek lisans	Gazi Üniversitesi / Bilgisayar Mühendisliği	Devam Ediyor
Lisans	Bilkent Üniversitesi / Bilgisayar Mühendisliği	2007
Lise	Beşiktaş Atatürk Anadolu Lisesi	2001

İş Deneyimi

Yıl	Yer	Görev
2008-Halen	Sahil Güvenlik Komutanlığı	Bilgi Güvenliği Şube Müdürü

Yabancı Dil

İngilizce

Yayınlar

Önder, M. (2019). Android zararlı yazılım tespit ve korunma çalışmaları. R. Karapınar ve T. Soldatovic (Editörler). *VI. Uluslararası Fen, Mühendislik ve Mimarlık Bilimlerinde Akademik Çalışmalar Sempozyumu* (345-358) içinde. Elazığ: Asos Yayınevi

Hobiler

Futbol, Sinema, Yazılım, Gezi

GAZİ GELECEKTİR..